MASARYKOVA UNIVERZITA

FAKULTA SPORTOVNÍCH STUDIÍ

Retrospektivní analýza ziskovosti sportovního sázení s pomocí strojového učení

Diplomová práce

BC. JAN PAŘÍZEK

Vedoucí práce: Mgr. Michal Bozděch, Ph.D.

Katedra tělesné výchovy a společenských věd
Program Management sportu

Brno 2024

MUNI
SPORT

Bibliografický záznam

Autor: Bc. Jan Pařízek
Fakulta sportovních studií
Masarykova univerzita
Katedra tělesné výchovy a společenských věd

Název práce: Retrospektivní analýza ziskovosti sportovního sázení s pomocí strojového učení

Studijní program: Management sportu

Vedoucí práce: Mgr. Michal Bozděch, Ph.D.

Rok: 2024

Počet stran: 94

Klíčová slova: umělá inteligence, strojové učení, hluboké učení, logistická regrese, rozhodovací strom, neuronová síť, sportovní sázení, výnosnost sportovního sázení

Bibliographic record

Author: Bc. Jan Pařízek
Faculty of Sports Studies
Masaryk University
Department of Physical Education and Social Sciences

Title of Thesis: Retrospective Analysis of Profitability in Sports Betting Using Machine Learning

Degree Programme: Sports Management

Supervisor: Mgr. Michal Bozděch, Ph.D.

Year: 2024

Number of Pages: 94

Keywords: artificial intelligence, machine learning, deep learning, logistic regression, decision tree, neural network, sports betting, sports betting yield

Anotace

Diplomová práce je zaměřena na využití umělé inteligence, a to konkrétně modelů strojového učení, v oblasti sportovního sázení. V první části výzkumu jsou natrénovány modely strojového učení na datech tenisových zápasů a následně je vybrán nejúspěšnější model na základě jeho prediktivní přesnosti. Dále jsou vytvořeny algoritmy pro jednotlivé zvolené sázkové strategie, na které se aplikují jak predikce vytvořené modelem strojového učení, tak i kurzy vypsané bookmakery. V závěru výzkumu je srovnána výdělečnost jednotlivých sázkových strategií s použitím vytvořených predikcí při komparaci s vypsanými kurzy od bookmakerů.

Abstract

The diploma thesis focuses on the utilization of artificial intelligence, specifically machine learning models, in the field of sports betting. In the first part of the research, machine learning models are trained on tennis match data, and subsequently, the most successful model is selected based on its predictive accuracy. Following this, algorithms are developed for each chosen betting strategy, to which both the predictions generated by the machine learning model and the odds provided by bookmakers are applied. Finally, the profitability of each betting strategy is compared using the created predictions in comparison with the odds provided by bookmakers.

Čestné prohlášení

Prohlašuji, že jsem diplomovou práci na téma **Retrospektivní analýza ziskovosti sportovního sázení s pomocí strojového učení** zpracoval sám**.** Veškeré prameny a zdroje informací, které jsem použil k sepsání této práce, byly citovány v textu a jsou uvedeny v seznamu použitých pramenů a literatury.

V Brně 24. dubna 2024

.....

Bc. Jan Pařízek

Poděkování

Chtěl bych poděkovat vedoucímu diplomové práce Mgr. Michalu Bozděchovi, Ph.D. za jeho ochotu, odborné vedení a cenné rady, které mi pomohly zkompletovat tuto práci.

Obsah

Seznam obrázků

Seznam tabulek

1 Úvod

Tato práce se zaměřuje na využití umělé inteligence, a to konkrétně strojového učení, v oblasti predikcí výsledků sportovních utkání a sportovního sázení.

Umělá inteligence má významný vliv na sportovní průmysl. Nejedná se pouze o zlepšování sportovního vybavení, sportovního vysílání, optimalizace tréninku a herních strategií, ale právě predikce výsledků sportovních utkání jsou ukázkovým příkladem využití strojového učení. Za předpokladu, že je k dispozici dostatek dat pro trénink samoučících modelů, můžeme predikce využít nejen pro lepší přípravu sportovců, ale i v oblasti sportovního sázení.

Teoretická část práce obsahuje uvedení do sportovního managementu – jak souvisí s využitím umělé inteligence pro predikci výsledků sportovních utkání a konsekventně se sportovním sázením. Dále je uvedena obecná charakteristika sportovního sázení, včetně jeho historie, makroekonomického dopadu na vybrané státy a etické stránky, ale primárně na představení vybraných sázkových strategií. Klíčová součást strojového učení je práce s daty a jeho pečlivá příprava pro aplikaci samoučících modelů. Proto je třetí kapitola věnována právě definici a charakteristice dat a oboru datové vědy, jejímž předmětem je manipulace s daty a jejich následná analýza. Poslední kapitola teoretické části se zabývá umělou inteligencí obecně, ale i konkrétně strojovým a hlubokým učením, pod které spadají samoučící modely využité pro výzkum v této práci.

Cílem výzkumu je v první řadě transformace dat tenisových zápasů tak, aby na ně bylo možné aplikovat modely strojového učení. Dále následuje trénování jednotlivých samoučících modelů a vyhodnocení prediktivně nejpřesnějšího modelu. Vytvořené predikce výsledků tenisových zápasů pak budou využity při retrospektivní simulaci sportovního sázení s využitím předem stanovených sázkových strategií pro analýzu jejich výnosnosti. Výnosnost strategií využívajících vytvořených predikcí bude komparována s výnosností strategií využívajících vypsaných kurzů od bookmakerů.

2 Syntéza poznatků

2.1 Sportovní management

Vymezit záběr sportovního managementu není jednoduchý úkol. Tento obor totiž zahrnuje nespočet činností. Pro základní rozlišení činností sportovního manažera můžeme jeho práci rozdělit do dvou skupin – práce s atlety a administrativní práce.

Práce s atlety může například zahrnovat role jako agent sportovců nebo jejich trenér či sportovní scout. Na druhou stranu administrativní práce zahrnuje činnosti ve sportovních organizacích, které umožňují realizaci sportu jako takového, a to jak na amatérské, tak profesionální úrovni. Mezi takové role můžeme řadit například manažery sportovišť, PR specialisty, marketing specialisty, anebo i sportovní datové analytiky (Srakocic, n.d.). A právě tato poslední role úzce souvisí s tématem této práce. Stejně jako v každém jiném průmyslu, i ve sportu dochází k obrovskému nárustu sběru dat. Data jsou velmi cennou surovinou pro jakéhokoliv manažera, který díky nim může dělat informovaná rozhodnutí. Používají se, mimo nespočet dalších případů, například ve fotbale pro volbu vhodného brankáře na danou fázi zápasu, či v NBA pro analýzu her a následnou optimalizaci coachingu (Pykes, 2022).

S rostoucím objemem dat je potřeba stále komplexnějších nástrojů, které je dokáží zpracovat a odpovědět skrze ně na důležité otázky – nejefektivnějším způsobem je zaručeně umělá inteligence, která celý proces analýzy dat automatizuje, a co by člověku pečlivou manuální analýzou trvalo dny až týdny, zvládne vytvořit během pár vteřin. Jednou z oblastí, kde je umělá inteligence hojně využívána, je predikce výsledků sportovních utkání, a tedy i konsekventně u sportovního sázení. Sportovní management v organizacích zabývajících se sportovním sázením těchto nástrojů mohou využívat například pro výpočet vhodných sázkových kurzů na jednotlivé subjekty sázky pro maximalizaci návratnosti. Naopak sportovní management v oblasti práce s atlety může využít predikcí průběhu a výsledků sportovních utkání pro lepší přípravu sportovců.

2.2 Sportovní sázení

2.2.1 Charakteristika sportovního sázení

V základních termínech by se sportovní sázení dalo popsat jako sázení peněz na výsledek sportovního utkání. Tradičním způsobem sázení je sázení proti bookmakerům, kteří vypisují kurz na jednotlivé možné výsledky. Pakliže si vsadím na výsledek, který nastane, dostanu od bookmakera peníze, které jsem vsadil vynásobené konkrétním

koeficientem kurzu. V případě, že si vsadím na výsledek, který nenastal, zůstanou mé peníze v rukou bookmakera.

2.2.2 Druhy sázkových kurzů

Běžně se vyskytují tři druhy vypsaných kurzů. V první řadě jsou to tzv. desetinné kurzy (také známe jako „evropské“). V rámci tenisu v tomto kurzovém systému můžeme například na hráče A vidět vypsaný kurz 1,5. V případě, že si na hráče A vsadíme a vyhrajeme, od bookmakera dostaneme náš vklad vynásobený daným kurzem – tedy při sázce 1.000 Kč vyhrajeme 1.500 Kč (1.000 * 1,5). Náš celkový profit pak získáme jednoduše odečtením vkladu od celkové výhry – v tomto případě tedy 500 Kč.

Dalším druhem kurzů, se kterými se ve sportu můžeme setkat, jsou tzv. zlomkové kurzy. Pokud bude na hráče A vypsaný například kurz $7/5$, náš výdělek v případě výhry vypočítáme vynásobením vkladu a čitatele zlomku a následného vydělení výsledku předchozí operace jmenovatelem zlomku kurzu. V praxi při sázce 1.000 Kč to bude vypadat následovně:

$$\textbf{\textit{Krok}}\ \mathbf{1}\text{:}\quad Vklad * \check{c}itatel = 1.000 * 7 = 7.000\ K\check{c}$$

$$\textbf{\textit{Krok}}\ \mathbf{2}\text{:}\quad \frac{V\acute{y}sledek\ kroku\ 1}{5} = \frac{7.000}{5} = 1.400\ K\check{c}$$

Čistý profit takové sázky je tedy 1.400 Kč, přičemž celkovou sumu obdrženou touto sázkou jednoduše vypočítáme sečtením původního vkladu a čistého zisku. Dalším hojně používaným druhem kurzů jsou tzv. americké kurzy. Americký kurz může mít buď záporné nebo kladné znaménko – pokud má znaménko záporné, uvádí to informaci, kolik musí člověk vsadit, aby vyhrál $100. Pokud je tedy kurz například -125, pak musíme vsadit $125, abychom při výhře vydělali $100. Naopak pokud je znaménko kladné, uvádí to, kolik vyhrajeme, pakliže vsadíme $100. V případě kladného kurzu 500 bychom při sázce $100 a následné výhře vydělali $500 (Sohail, 2024).

2.2.3 Historie sportovního sázení

Gambling pravděpodobně předchází zaznamenané historii – první zmínky o sázení a losech jsou již v biblických příbězích. Organizované sázení se objevilo s příchodem organizovaného sportu – nejznámějším případem toho jsou bezpochyby antické olympijské hry. Ty sahají až do roku 776 př.n.l. a trvaly do roku 394 n.l. Mimo Řecko jsou to pak určitě závodní a bojové sporty v antickém Římě – jmenovitě závodní kočáry a gladiátorské zápasy (Matheson, 2021). Všechny tyto události vytvořily místo na trhu pro organizované sázení.

Ovšem stejně jako dnes máme kolem sázení a gamblingu určité stigma, i v antických časech byly zaznamenány všemožné problémy spojené s takovými aktivitami. Mezi první zaznamenané historické skandály se řadí antický boxer Eupolus z Thesálie, který měl údajně podplácet své soupeře, aby proti němu záměrně prohráli. Dalším obdobným případem je římský císař Nero, který se účastnil Olympijských her v roce 67 n.l. a podplatil rozhodčí astronomickými částkami, aby do her například přidali jím zvolené, velmi specifické, disciplíny, anebo při závodech udělili císaři první místo i v případech, kdy třeba závod sám vůbec nedokončil. Dohromady si Nero ze všech možných Olympiád a jiných řeckých sportovních závodů odnesl 1808 odměn za první místo (C. Klein, 2021).

Z důvodu etických a ekonomických rizik sázení a gamblingu obecně dokonce římský císař Augustus omezil tyto aktivity pouze na jeden týden v roce, a to během svátků Saturnálie. Opakem toho jsou pak císaři, kteří těmto aktivitám propadli – například císař Commodus, který císařský palác v Římě proměnil v kasino a v procesu zbankrotoval celou římskou říši (Matheson, 2021).

Dokud v polovině devatenáctého století nezačaly vznikat různé sportovní ligy napříč Evropou a Severní Amerikou, v období renesance a před industriální revolucí byly hlavním předmětem sázení koňské dostihy. V sedmnáctém století byly dostihy navštěvovány britským králem Karlem II tak často, že se tento sport začal nazývat „sportem králů“. První závodní okruh pro koňské závody v „novém světě“ pak byl založen v roce 1665 na Long Islandu, a od té doby jsou dostihy v Americe populárním sportem do dnes.

Nejrozšířenějším druhem vyplácení výher bylo do poloviny devatenáctého století jednoduché vypisování kurzů na výhru na jednotlivé subjekty závodu, na základě kterých poté byly vyplaceny odměny. Takový způsob sázení však nese výrazná rizika jak pro bookmakery, tak pro sázkaře – může se stát, že bookmaker musí vyplatit velkou sumu peněz a ve výsledku bude prodělávat, a na druhou stranu sázkaři hrozí riziko, že bookmaker jednoduše nemá na vyplacení všech výher (Matheson, 2021).

Tento problém vyřešil v roce 1867 Joseph Oller (mimo jiné i zakladatel pařížského nočního klubu Moulin Rouge), který přišel s návrhem tzv. pari-mutuel sázení. Tento způsob nevyplácí výhry na základě nezávisle vypsaných kurzů bookmakerem, ale namísto toho jsou výhry vypláceny z peněž, které byly všemi sázkaři zaplaceny – jedná se tedy o stejný systém, který se dnes nachází ve finanční matematice a pojišťovnictví.

O téměř 130 let později, v roce 1995, byly založeny první sázkové webové stránky, které popularitu sportovního sázení výrazně podpořily. Dodnes se v online prostředí hojně využívá jak klasického vypisování kurzů, tak i pari-mutuel sázení. V roce 2000 však web Betfair přišel s revoluční myšlenkou – Betfair nebral sázky od svých uživatelů, ale nechal je sázet mezi sebou (tzv. sportovní loterie) (Charpentier, 2019).

2.2.4 Makroekonomická stránka sportovního sázení

Pro představu, jaký má legální sportovní sázení vliv na HDP státu, se blíže zaměříme na USA. V roce 2018 případ Nejvyššího soudu „Murphy v. National Collegiate Athletic Association“ vyústil ve zrušení zákona o ochraně profesionálního a amatérského sportu z roku 1992, který zakázal téměř všechny státem povolené sportovní hazardní hry po celých Spojených státech; do té doby bylo sportovní sázení legální pouze ve třech státech – v Nevadě byl povolen veškerý sportovní gambling, a v Oregonu a Delaware pouze sportovní loterie (Pempus, 2024). V roce 2023 sportovní gambling zlegalizovalo plně 22 států, 9 států má povoleno pouze retail sázení (tedy pouze na pobočce, nikoliv online) a ve dvou se legálně může sázet pouze online (Ardeljan, 2023). Ve článku Emilian Ardeljan také uvádí, že díky legalizaci sportovního sázení se ve státech s plnou legalizací zvýšilo HDP per capita o $2.800, zatímco ve státech, které zlegalizovali pouze online sázení došlo ke zvýšení HDP per capita o $1.100 (Ardeljan, 2023). Při posuzování legalizace např. právě sportovního sázení je potřeba samozřejmě zohledňovat i faktory nefinančního charakteru – například vliv sázení na lidi s mentálními poruchami apod. – ale z ekonomické stránky věci lze určitě říct, že sportovní sázení má pozitivní vliv na makroekonomii státu.

Z evropských zemí v roce 2019 má gambling podle webu Statista (2020) nejvyšší podíl na HDP v Řecku (1,16 %, tedy $2,38 miliardy z celkového HDP $205,3 miliardy). Hned za ním je Itálie, Portugalsko a Finsko (1,02 %, 0,89 %, 0,85 %). Naopak nejnižší zastoupení na HDP má Polsko (0,3 %, tedy $1,79 miliardy z celkového HDP $596,1 miliardy), a dále Nizozemsko, Rakousko a Německo (0,38 %, 0,4 %, 0,42 %).

2.2.5 Přehled sázkových strategií

Ucelená sázková strategie stanovuje sázkaři dvě primární věci – na který zápas vsadit, a jakou částku. V této části práce jsou uvedeny známé kasinové sázkové strategie, které byly původně určeny pro ruletu či blackjack, nicméně se jich dá využít i ve sportovním sázení.

2.2.5.1 Martingale

Sázkový systém Martingale je jeden z nejstarších a nejznámějších sázkových strategií, který byl vytvořen pro ruletu a blackjack. Jedná se tedy o strategii, která je populární především v situaci, kdy se vsází na binární výsledek (tedy výhra či prohra, červená nebo černá apod.). Hlavní myšlenkou stojící za Martingale systémem je získat zpět ztráty z prohry tím, že po každé porážce zdvojnásobíme velikost naší sázky, abychom eventuálně peníze získali zpátky. Průběh sázení podle této strategie může být ilustrován čtyřmi jednoduchými kroky:

1. V první řadě je zvolena velikost počáteční sázky – pro tento příklad si stanovme například 100 Kč.
2. Po každé výhře bude následující sázka opět ve výši té počáteční. Pokud se nám tedy podaří vyhrát, opět sázíme ve výši 100 Kč.
3. Pakliže sázku prohrajeme, při další sázce naši sázku zdvojnásobíme. Pokud tedy vsadíme počáteční výši 100 Kč a prohrajeme, na další hru (nebo ve sportu zápas) vsadíme 200 Kč. Pokud prohrajeme podruhé, v další sázce vsázíme 400 Kč atd.
4. Jakmile se nám po prohrách podaří vyhrát, v další sázce se opět stáhneme na naši zvolenou počáteční výši sázky – například tedy po dvou prohrách z 400 Kč zpátky na 100 Kč a celý tento proces opakujeme.

Myšlenkou tedy je, že eventuálně při výhře získáme zpátky všechny peníze, které jsme prohráli – pokud to spojíme s faktem, že například při 50/50 sázení máme propenzitu polovinu sázek vyhrát, může to být teoreticky téměř neprůstřelná strategie, se kterou může konzistentně generovat profit (Tuček & Dolinová, 2001).

To však neznamená, že tomu tak v praxi skutečně je. Ačkoliv se kterýkoliv systém může v teoretické rovině jevit jako nenapadnutelný, v reálném světě narazíme na mnoho překážek. První takovou překážkou je fakt, že kasina a online sázecí platformy omezují maximální možnou výši sázky – nejen že je vždy přítomné nějaké výchozí omezení pro všechny, ale hráči/sázkaři, kteří očividně používají sofistikované strategie, díky kterým konzistentně více peněž vyhrají než prohrají, jsou kasiny a sázecími platformami penalizováni ve formě přísnějších omezení. Velké riziko u Martingale systému je, že můžeme narazit na mnoho proher v sérii, a pakliže nám omezení maximální sázky zamezí v sázení dalšího dvojnásobku, funkčnost celého systému rázem padá. Do stejné situace se můžeme dostat i z důvodu omezeného osobního kapitálu, který máme vyhrazen pro sázení (Tuček & Dolinová, 2001).

2.2.5.2 Great Martingale

Great Martingale je variací systému Martingale. V zásadě se jedná o stejný systém s tím rozdílem, že sázku po prohře nejenom dvojnásobíme, ale zároveň k ní připočteme námi zvolenou jednotku navíc. Princip tedy zůstává stejný – získat zpátky prohrané peníze dvojnásobením sázek, ale celý proces urychlit přidáním peněžní jednotky ke každé prohrané sázce (Tuček & Dolinová, 2001).

Pokud tedy jako počáteční sázku zvolíme opět 100 Kč a 10 Kč jednotku, pak při prohře vsadíme nejen dvojnásobek 100 Kč (tedy 200 Kč), ale přičteme k tomu i zvolenou jednotku 10 Kč – celková vsazená suma pak činí 210 Kč. Pakliže bychom prohráli i podruhé, zdvojnásobíme předchozí sázku (tedy 210 * 2 = 420 Kč) a navíc přičteme zvolenou jednotku – celková další sázka tedy bude 430 Kč. Tento proces se opakuje při každé další prohře.

Nicméně Great Martingale systém trpí stejnými riziky, které jsem již zmiňoval u výchozího Martingale systému, a navíc zvyšujeme riziko, že při sérii proher a omezeného kapitálu potenciálně rychleji přijdeme o naše finance.

2.2.5.3 Anti-martingale

Tato variace Martingale systému funguje téměř identicky s výchozím Martingale systémem, pouze s rozdílem, že výši sázky zdvojnásobujeme při výhře, nikoliv prohře. Pakliže tedy původní sázku zvolíme opět na 100 Kč, při každé následující výhře sázku zdvojnásobíme (200, 400, 800 atd.), přičemž při každé prohře se vracíme zpět na původní výši sázky (100 Kč) (Tuček & Dolinová, 2001).

2.2.5.4 Fibonacciho systém

Fibonacciho sekvence je pojmenována po italském matematikovi Fibonaccim. Ten ji západnímu světu představil ve 13. století n. l. Nicméně důkazy nasvědčují tomu, že byla již objevena v Indii již ve druhém století př. n. l. Tato sekvence je unikátní tím, že se přirozeně vyskytuje nejen v matematice, ale i v přírodě – například se s ní můžeme setkat v uspořádání větvení stromů nebo jejich listů (Oddspedia, 2023).

První dvě čísla v sekvenci jsou vždy 0 a 1. Následující čísla jsou pak vždy součtem dvou předchozích. Matematicky bychom to mohli vyjádřit následovně:

$$x_1 = 0;\ x_2 = 1;\ x_n = x_{n-1} + x_{n-2}$$

Budeme-li se tedy řídit této rovnice, prvních pár čísel v sekvenci bude vypadat následovně:

0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, 233, 377, 610 …

Co se týče této sekvence při sázení, stejně jako u ostatních systémů si musíme zvolit počáteční finanční jednotku – držme se příkladu 100 Kč. V sekvenci pak pouze násobíme tuto libovolnou jednotku konkrétními čísly v sekvenci, pouze s tou výjimkou, že x_1 (tedy počáteční nulu v sekvenci) vynecháme. Ukažme si tedy sekvenci na prvních sedmi číslech:

100, 100, 200, 300, 500, 800, 1300 …

Samotná sázková strategie pak tkví ve dvou krocích. Při prohře se na sekvenci posuneme o jedno číslo doprava a tuto částku sázíme v další sázce. Při výhře se naopak posouváme doleva, a to tentokrát o dvě čísla. Pro ukázku si vytvořme sérii tří proher a následujících dvou výher:

- **Prohra 1:** ze 100 Kč (pozice 1) na 100 Kč (pozice 2)
- **Prohra 2:** ze 100 Kč (pozice 2) na 200 Kč (pozice 3)

- **Prohra 3:** z 200 Kč (pozice 3) na 300 Kč (pozice 4)
- **Výhra 1:** ze 300 Kč (pozice 4) na 100 Kč (pozice 2)
- **Výhra 2:** ze 100 Kč (pozice 2) na 100 Kč (pozice 1) – jelikož se v tomto kroku již nelze posunout o dvě čísla (pozice) doleva, jednoduše sázíme 100 Kč z první pozice sekvence.

2.2.5.5 D'Alembert systém

Tento systém opět funguje na podobný princip jako ty předchozí – prvním krokem je zvolit si výchozí finanční jednotku, kterou budeme sázet. Následně při každé prohře zvyšujeme následující sázku o jednu tuto jednotku, a naopak při každé výhře následující sázku snižujeme a jednu jednotku (Tuček & Dolinová, 2001).

Pokud tedy začínáme sázkou 100 Kč a prohrajeme, další sázku zvýšíme na 200 Kč. Pakliže vyhrajeme a dostaneme se zpět na 100 Kč (snížíme po výhře sázku o jednu jednotku), pak v případě, že následuje druhá výhra po sobě, již výši sázky nesnižujeme a vsázíme opět 100 Kč – jinými slovy pod námi stanovenou finanční jednotku se „nesmíme" dostat.

2.2.6 Omezení a rizika negativní progrese při sázení

Většina systémů zmíněných v této kapitole se řídí principem tzv. „negativní progrese" – progrese, ať už pozitivní či negativní, jednoduše znamená, že při konkrétní variantě binárního výsledku zvýšíme či snížíme výši následující sázky. Pokud se zabýváme například ruletou, kde je pravděpodobnost výhry/prohry vždy 50/50, je statisticky prakticky zaručeno, že se počet výher a proher bude po dostatečně dlouhém časovém intervalu sobě (téměř) rovnat. Pro ilustraci byla vytvořena simulace sekvence tisíce 50/50 her v programovacím jazyce Python, která ukáže poměr výstupu výher a proher:

```
from random import randint

i_lst = []
for i in range(5):
    x, y = 0, 0
    for j in range(1000):
        num = randint(0,1)
        if num == 0:
            x += 1
        else:
            y += 1
    i_lst.append([x,y])

print(i_lst)
```

Jelikož sekvenci tisíce her simulujeme pětkrát za sebou, výstupem tohoto kódu bude následující dvourozměrná matice (při každém novém spuštění kódu se čísla samozřejmě budou mírně lišit, ale účelem je nyní ukázat průměrnou hodnotu rozdílu mezi prohrami a výhrami):

```
[[490, 510], [506, 494], [504, 496], [488, 512], [485, 515]]
```

V každém z vygenerovaných pěti výsledků máme list o dvou číslech – prvním je počet proher a druhým počet výher z odehraných tisíce her. V dalším kroku je na řadě simulace Martingale systému, opět na tisíci hrách, s výchozí výší sázky 100 Kč.

```
from random import randint

profit_lst = []
for i in range(20):
    betting_unit = 100
    profit = 0
    for j in range(1000):
        num = randint(0,1)
        if num == 0:
            profit -= betting_unit
            betting_unit *= 2
        else:
            profit += (betting_unit*2)-betting_unit
            betting_unit = 100

    profit_lst.append(round(profit))
print(profit_lst)
```

Výstupem této simulace je konečný profit po tisíci sázkách držících se principů Martingale systému. Jelikož tuto simulaci spouštíme dvacetkrát za sebou, výsledkem je dvacet čísel znázorňujících konečný profit (v Kč) po vsazení celkem 1,000,000 Kč:

```
[50800, 48300, 49400, 53400, 49100, 50600, 48700, 51500, 51100, 47700, 49400,
50000, 50100, 50200, 50200, 48900, 48600, 49900, 49100, 48600]
```

A přesně na tomto výsledku lze vidět hlavní premisu sázení s negativní progresí. Jelikož je statisticky zaručeno víceméně 50% výher a 50% proher na dostatečně dlouhé časové ose a všechny ztracené peníze ze sekvence proher se vrátí první následující výhrou, pak je prakticky získán pouze profit jak z dvojnásobku před první následující výhrou, tak ze sekvencí výher – jinými slovy je statisticky vysoce nepravděpodobné, že se člověk dostane na záporné číslo.

```
from random import randint

x = 0
y = 0
for i in range(10000):
    betting_unit = 100
    profit = 0
    for j in range(1000):
        num = randint(0,1)
        if num == 0:
            profit -= betting_unit
            betting_unit *= 2
        else:
            profit += (betting_unit*2)-betting_unit
            betting_unit = 100

    if profit < 0:
        x += 1
    else:
        y += 1
print([x,y])
```

Kód výše simuluje sekvenci tisíce her s dodržování pravidel Martingale systému, ale tuto sekvenci opakujeme deset tisíckrát za sebou. Výsledným listem jsou dvě čísla – první zastupuje počet sekvencí, které skončili finanční ztrátou; druhé zastupuje sekvence, které skončili kladným profitem:

```
[18, 9982]
```

Ze simulace tedy vychází, že pravděpodobnost, že Martingale systém (při dotaci tisíce her) skončí ve ztrátě, je mizivých 0,18 %, přičemž pravděpodobnost zisku je 99,82%. Na teoretické rovině tyto systémy tedy skutečně fungují. Ale jak již bylo zmíněno, v reálném světě nemáme neomezené prostředky na sázení. Pokud tedy narazíme na sekvenci např. deseti proher za sebou, náš požadavek na další sázku se exponenciálně zvyšuje a pravděpodobně dříve či později vyčerpáme všechen náš kapitál.

Při sázení je tedy potřeba si vytvořit finanční plán, ve kterém zohledníme naše finanční omezení, omezení maximální možné sázky umožněné zvoleným poskytovatelem sázek, a ideálně si nasimulovat pravděpodobnost, že na sázení budeme v dlouhodobém horizontu vydělávat. Nechat se zlákat sázkovými systémy čistě na základě teoretické funkčnosti může být lákavé, ale zároveň velice zrádné.

2.2.7 Etická stránka gamblingu

Etika gamblingu je komplexní problematika, která se prolíná s různými morálními, sociálními a ekonomickými úvahami. Na jedné straně zastánci tvrdí, že jednotlivci mají autonomii při rozhodování o tom, jak utratí své peníze, včetně zapojení do činností, jako jsou například hazardní hry a gambling obecně a že zodpovědné hraní může být formou zábavy srovnatelnou s jinými rekreačními aktivitami.

Etika gamblingu se však stává spornější při zkoumání problémů, jako je závislost, vykořisťování a marketingové praktiky v tomto odvětví. Kritici zdůrazňují obavy z potenciální škody, kterou může hazardní hraní způsobit jednotlivcům a komunitám. Zejména závislost na hazardních hrách je vážným etickým problémem, protože může vést k finančnímu krachu, napjatým vztahům a celkově negativním dopadům na duševní zdraví (Chóliz, 2018).

2.3 Data a datová věda

Digitální informace se v nedávné minulosti začaly označovat za „surovinu zítřka“ (Birch et al., 2021). V dnešní době však takové označení není v žádném případě nadsázkou. Výrazné navýšení sběru dat velkými technologickými společnostmi (označovány jako „Big Tech“) umožnilo vytváření sofistikovanějších technologií a algoritmů, bez kterých by si nikdo z nás už nedokázal představit žít. Za účelem ilustrace výrazného nárůstu kvantitativního rozsahu užívání a sběru dat v průběhu poslední dekády, se zaměřme na rok 2010, kdy byla zaznamenána produkce přibližně 2 zettabytů (ZB) digitálních informací. Pokud bychom tyto informace reprezentovali klasickými flash disky o velikosti 1 GB a uspořádali je za sebou, dosahovala by celková délka této sestavy zhruba 184 milionů standardních fotbalových hřišť. Překvapivě, pouhých deset let později, v roce 2020, dosáhla produkce digitálních informací přes 44 zettabytů (Saha, 2020).

Jak je ale tak drastický nárůst možný za tak krátký čas? Většina těchto digitálních informací se skládá z osobních dat uživatelů všemožných zařízení – jmenovitě hlavně chytré telefony, různé senzory (chytré hodinky apod.), automobily s připojením k internetu apod. Tyto technologie se za posledních 15 let výrazně zlepšily a hlavně se staly dostupnými většině lidem na planetě – uživatelé využívající těchto zařízení a různých aplikací na nich jsou tedy pro Big Tech společnosti „zlatým dolem“. Není totiž náhodou, že hodnota a moc těchto společností v tomto století natolik vzrostly – britský filozof Francis Bacon (1561-1626) pronesl svůj slavný výrok „vědění je moc“ (J. Klein & Giglioni, 2020), který se dá perfektně aplikovat právě na tuto situaci. Čím více dat o něčem máme, tím více o tom víme, popřípadě o tom můžeme usoudit. A přesně na základě toho vznikl obor datové vědy („data science“).

2.3.1 Definice dat

Data bychom mohli popsat jako matici $\boldsymbol{r}$ x $\boldsymbol{s}$ o $\boldsymbol{r}$ řádcích a $\boldsymbol{s}$ sloupcích. Každý jednotlivý řádek zastupuje konkrétní objekt, zatímco sloupce představují atributy (či vlastnosti) těchto objektů.

Tab. 1: Příklad základní podoby dat

id	věk	pohlaví	váha	výška
0	21	muž	87	187
1	28	žena	56	170
2	35	muž	82	183
3	19	muž	75	179
4	25	žena	59	173
5	31	muž	76	176
6	22	žena	52	165

Tabulka č. 1 reprezentuje příklad možné podoby datasetu. První sloupec v jakýchkoliv datech bývá většinou zpravidla tzv. „indexový" – ten stojí za unikátností daného objektu na daném řádku. V tomto případě máme jako první sloupec „id", který je na každém řádku unikátní (tzn. nemůže se vyskytovat v datech dvakrát či vícekrát na různých řádcích). Tato konvence slouží hlavně pro efektivní vyhledávání v databázích. Pokud hledáme konkrétní osobu, pak se záběr samozřejmě dá zúžit hledáním například podle výšky, ale efektivnější je hledat podle něčeho unikátního pro danou osobu. I například spojení křestního jména a příjmení může být duplicitní (speciálně v databázích obrovských rozměrů – například na sociálních sítích jako Facebook). Proto je vhodné vždy u objektu držet nějaké unikátní identifikační číslo nebo kód, podle kterého vždy najdeme pouze tu instanci, kterou hledáme, a nic jiného.

Ostatní sloupce pak označujeme jednoduše za „datové". Tyto sloupce již představují informace o daném objektu – v případě tabulky č. 1 jsou ke každému objektu uvedeny čtyři datové sloupce, a to informace o věku, pohlaví, váze a výšce daných osob. Datové sloupce však již mohou být duplicitní (tzn. hodnoty mohou být stejné na dvou či více řádcích), protože pouze popisují vlastnosti objektu a neoznačují tak objekt samotný.

Data však mohou mít i komplikovanější podoby. Příkladem mohou být obrázky. Lidským okem lze rozpoznat, co je obsahem obrázku, ale aby s obrázkem pracoval počítač, musí být transformován do matice stejně, jako jsou uvedeny například atributy jednotlivých lidí v tabulce č. 1. Avšak v tomto případě místo věku, pohlaví apod. budou v matici informace o jednotlivých pixelech.

2.3.2 Jednorozměrná a vícerozměrná data

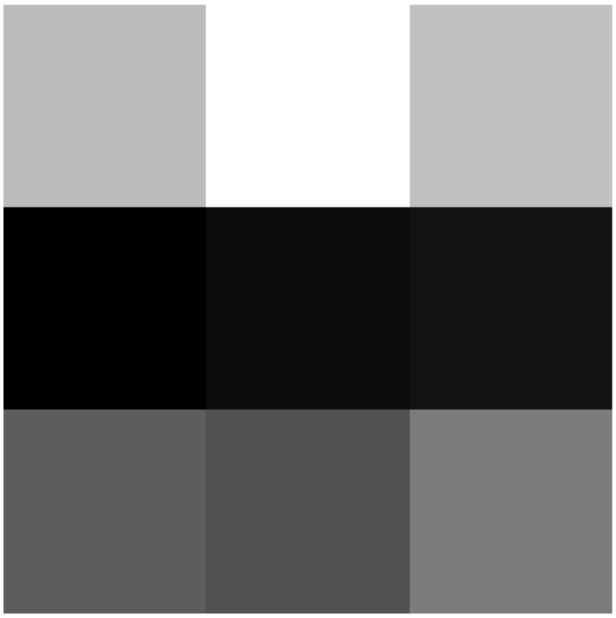
Nejjednodušší formou dat jsou data jednorozměrná. V programovacích jazycích se jedná o primitivní datový typ, který slouží především pro ukládání jednoduchých číselných řad. V reálném světě mohou být například využity pro informaci o průměrné teplotě za posledních sedm dní následovně:

```
average_weather = [21, 23, 24, 22, 21, 25, 27]
```

Z vícerozměrných dat jsou nejvíce využívané dvou a troj-rozměrná data. Obě tyto varianty se dají jednoduše vysvětlit na obrazových datech zmíněných v předchozí podkapitole.

Představme si černobílý obrázek – pro jednoduchost řekněme, že má rozlišení 3x3 pixely, tudíž dohromady obsahuje pouze 9 pixelů. Každý z těchto pixelů obsahuje určitý stupeň šedi (v rozmezí 0–255, přičemž 0 je černá a 255 je bílá barva). Pokud bychom chtěli takový obrázek znázornit v programovacím jazyce (například v Pythonu), vytvoříme dvourozměrný list následujícím způsobem (generace obrázku č. 1 byla provedena přes *Python* knihovnu *matplotlib*):

```
grayscale_image = [[200, 255, 205], [50, 60, 65], [125, 115, 150]]
```

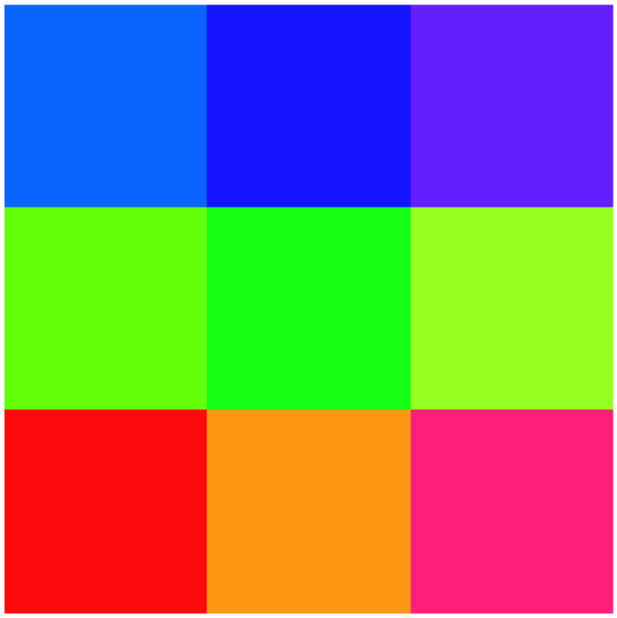


Obr. 1: Generace černobílého obrázku

Vnější hranaté závorky ohraničují obrázek jako celek. Vnitřní závorky znázorňují jednotlivé řádky, přičemž každý obsahuje přesně tři pixely konkrétního stupně šedi.

U barevných obrázků je nutné přidat další rozměr. Jedná se tedy o trojrozměrná data. U každého pixelu pak nestačí pouze odstín šedi, ale tři hodnoty zastupující RGB hodnotu pixelu. V Pythonu by takový obrázek (3x3 pixely) byl znázorněn následovně (generace obrázku č. 2 byla provedena přes *Python* knihovnu *matplotlib*):

```
color_image =
[[[10,100,255],[20,20,255],[100,30,255]],
[[100,255,10],[20,255,20],[150,255,30]],
[[255,10,10],[255,150,20],[255,30,120]]]
```



Obr. 2: Generace barevného obrázku

2.3.3 Atributy dat

Jak již bylo uvedeno, nezáleží na tom, zda jsou data uložena v tabulkách nebo v matici, vždy jsou přítomné jednotlivé objekty (řádky) a jejich atributy (sloupce). A ačkoliv to zatím nebylo předmětem rozboru, na příkladech dat výše bylo představeno několik druhů atributů. Nejjednodušším rozlišením atributů je na atributy numerické a textové.

Jak numerické, tak textové atributy mohou mít určité specifikace či omezení. Například textový údaj může či nemusí nabývat pouze omezeného počtu znaků (příklad „muž“ a „žena“ znázorněno písmeny „M“ a „Ž“ je omezeno pouze na jeden znak). Numerické údaje mohou být znázorněny v konkrétních jednotkách (u výšky například centimetry nebo u teploty libovolné stupně), a je opět možné omezit jejich rozsah – například můžeme stanovit maximální možnou výšku člověka nebo maximální možnou naměřenou teplotu.

2.3.3.1 Numerické atributy

Numerický (nebo také kardinální) atribut je vyjádřen reálnou číselnou hodnotou. Dělí se na dvě základní škály (Zaki & Meira, 2014):

- **Intervalová škála:** pro atributy na této škále můžeme použít pouze operace jako sčítání a odčítání. Mezi jednotlivými hodnotami jsou rovné intervaly a můžeme hodnoty smysluplně seřazovat. Na intervalové škále však nemáme skutečnou nulovou hodnotu. Příkladem toho se dá uvést teplota – mezi 10 °C a 20 °C je stejný

rozdíl jako mezi 20°C a 30°C, ale pokud je teplota 0°C, nedá se říct, že se jedná o absenci teploty. Proto ani nemůžeme říct, že 20 °C je „dvakrát“ více než 10°C.

- **Poměrová škála:** atributy na poměrové škále mají mezi hodnotami taktéž rovné intervaly, můžeme je smysluplně řadit, ale jejich klíčovou vlastností je, že mají skutečnou nulovou hodnotu. Nulová hodnota vyjadřuje absenci měřené hodnoty, a proto je například váha příkladem hodnoty na poměrové škále. 0 kg vyjadřuje „absenci váhy“, přičemž můžeme říct, že 20 kg je dvakrát více než 10 kg.

Dále můžeme numerické atributy dělit na tři skupiny:

- **Diskrétní čísla:** diskrétní čísla jsou od sebe odlišná a jasně oddělená – často reprezentují počitatelné objekty nebo celá čísla. Jako příklad si můžeme uvést například počet studentů ve třídě, počet aut na parkovišti apod.
- **Kontinuální čísla:** kontinuální čísla mohou zastupovat jakoukoliv hodnotu v daném rozsahu. Nejsou tedy omezena pouze na celá čísla, ale mohou nabývat i desetinných míst. Pro příklad si můžeme uvést váhu, výšku, teplotu či čas. Všechny tyto příklady mohou teoreticky obsahovat nekonečně dlouhé množství desetinných míst.
- **Dichotomická čísla:** dichotomická (neboli binární) čísla jsou speciálním druhem diskrétních čísel, které mohou nabývat pouze dvou konkrétních hodnot. Využití tak logicky naleznou v případech, kdy se očekávají pouze dva výsledky, například výhra/prohra, ano/ne, 0/1 apod.

2.3.3.2 Kategorické atributy

Kategorické atributy mají předem stanovené hodnoty. Může se jednat o například o pohlaví (muž, žena), spokojenost zákazníka (nízká, střední, vysoká) apod. Dělí se na dva typy (Zaki & Meira, 2014):

- **Nominální:** hodnoty nominálních kategorií jsou od sebe odlišné, ale nemají žádné inherentní logické seřazení. Příkladem mohou být barvy (červená, modrá, zelená…), pohlaví (muž, žena) nebo typy motorových vozidel (auto, motorka…). Všechny tyto kategorie mají předurčené hodnoty, které mezi sebou nemůžeme porovnávat na základě nějaké kvality – například nemůžeme říct, že modrá barva je objektivně lepší než červená apod.
- **Ordinální:** hodnoty ordinálních kategorií mají na rozdíl od nominálních inherentně logické seřazení, avšak intervaly mezi jednotlivými kategoriemi nejsou sobě rovné nebo jinak jasně definované. Například můžeme inženýrský stupeň vzdělání označit za „lepší“ než bakalářský, ale interval mezi těmito dvěma stupni není tak zřejmý jako například mezi čísly 1 a 2. Mimo dosažený stupeň vzdělání bychom za ordinální hodnoty mohli označit i socioekonomický status člověka/rodiny, již zmíněnou spokojenost zákazníků libovolného podniku apod.

2.3.3.3 Závislé a nezávislé proměnné

Poslední rozdělení atributů, které je pro téma této práce důležité, a budeme jej hojně využívat z pozdějších kapitolách, je rozdělení na závislé a nezávislé proměnné (Bhandari, 2022).

- **Závislé proměnné:** závislé proměnné představují výsledek nebo odpověď, který se ve výzkumu měří. Nazývají se závislé, protože jejich hodnota závisí na proměnných nezávislých – dochází tedy ke studiu toho, jak se závislé proměnné mění při změně hodnot proměnných nezávislých. Například se dá měřit, jak různé dávkování léků na krevní tlak ovlivňuje zdravotní stav pacienta, anebo pokud se budeme držet tématiky sportovního sázení, tak na kolik například výška, váha či věk ovlivňují pravděpodobnost, že daný sportovec vyhraje.
- **Nezávislé proměnné:** jak již bylo řečeno, s nezávislými proměnnými se manipuluje, aby se odhalil jejich efekt na závislou proměnnou. Považují se tedy za „příčinu“ hodnoty či stavu závislých proměnných.

2.3.4 Struktura dat

Jak jsme si již v minulých kapitolách ukázali, data můžeme skladovat v mnoha různých podobách či strukturách. Například můžeme mít pouze matici vyjádřenou na jednom řádku kódu, nebo můžeme využít tabulkových softwarů jako je např. Microsoft Excel nebo Google Sheets. V listu jsou uvedeny čtyři primární způsoby, jakými mohou být data strukturována (Dietrich et al., 2015).

- **Strukturovaná data**: tento typ struktury je asi první, který se každému vybaví – jedná se například právě o tabulky, kde jsou jasně definované jednotlivé řádky a sloupce (například soubory s koncovkou csv, xlsx, nebo transakční data).
- **Polo-strukturovaná data**: textová data se zřetelnou strukturou, která nám umožňují metodu parsování (rozeznání a úprava například XML souborů).
- **Kvazi-strukturovaná data**: textová data s nepravidelným formátem, které jde formátovat a seřadit s úsilím a správnými nástroji. Jedná se například o „web clickstream“ data – záznam například toho, na co uživatel v internetovém prohlížeči kliká apod.
- **Nestrukturovaná data**: Zde se jedná o data jako například obrázky, videa nebo textové soubory (např. PDF), které nemají žádnou inherentní strukturu.

2.3.5 Definice datové vědy

Datová věda se často označuje za koncept, který sjednocuje statistiku, datovou analýzu, a všechny jim příbuzné metody, za účelem odhalit a pochopit různé fenomény skrze data. Datová věda tedy jinými slovy studuje data, a jejím produktem jsou nové objevy, predikce, služby, doporučení, pomoc při důležitém rozhodnutí apod. (Sarker, 2021a) Nicméně ačkoliv se datová věda obecně dá shrnout touto „zjednodušenou“ definicí, ve

skutečnosti se stále vedou spory, co přesně je součástí datové vědy, jaké jsou úkoly a kompetence datových vědců apod. Způsobů, jak s daty pracovat, je obrovské množství, a každý zahrnuje nepřeberné množství různých nástrojů, kterými daný úkol můžeme zpracovat.

2.3.6 Historie datové vědy

Datová věda se dá z historického hlediska popsat jako propojení statistiky jakožto „vyspělé“ vědní disciplíny s vědní disciplínou výrazně novější, a to konkrétně informatikou (anglicky „computer science“).

Americký statistik John W. Tukey ve svém článku „The Future of Data Analytics“ z roku 1962 napsal výrok, v němž mohou být nalezeny zárodky vzniku datové vědy jakožto vědní disciplíny:

> *„Po dlouhou dobu jsem si myslel, že jsem statistik, zaměřený na inference z konkrétních případů na obecné. Ale sledováním vývoje matematické statistiky jsem začal pochybovat. Začal jsem si uvědomovat, že mé stěžejní zájmy směřují k analýze dat. Analýza dat a části statistiky s ní spojené musí přijmout charakteristiky vědy spíše než matematiky. Analýza dat je inherentně empirickou vědou. Jak důležitý a významný je vzestup elektronického počítače? V hodně případech by odpověď mohla překvapit mnohé tím, že je ‚důležitý, ale ne nezbytný,‘ ačkoli v jiných případech není pochyb o tom, že počítač je zcela ‚nezbytný‘.“ (Tukey, 1962)*

Nicméně až v roce 1996 je název „data science“ použit poprvé v názvu konference “Data science, classification, and related methods” v Japonsku. V roce 1997 pak profesor C. F. Jeff Wu z Michiganské Univerzity navrhoval přejmenování statistiky na datovou vědu a statistiků na datové vědce (Press, 2013). Nový svět informačních technologií a neustálé zvyšování objemu sbíraných dat v digitální podobě totiž představovalo pro dosavadní statistické metody problém. Profesor oboru dolování dat („data mining“) z Tel Aviv Univerzity Jacob Zahavi je citován ve článku „Mining Data for Nuggets of Knowledge“, kde říká:

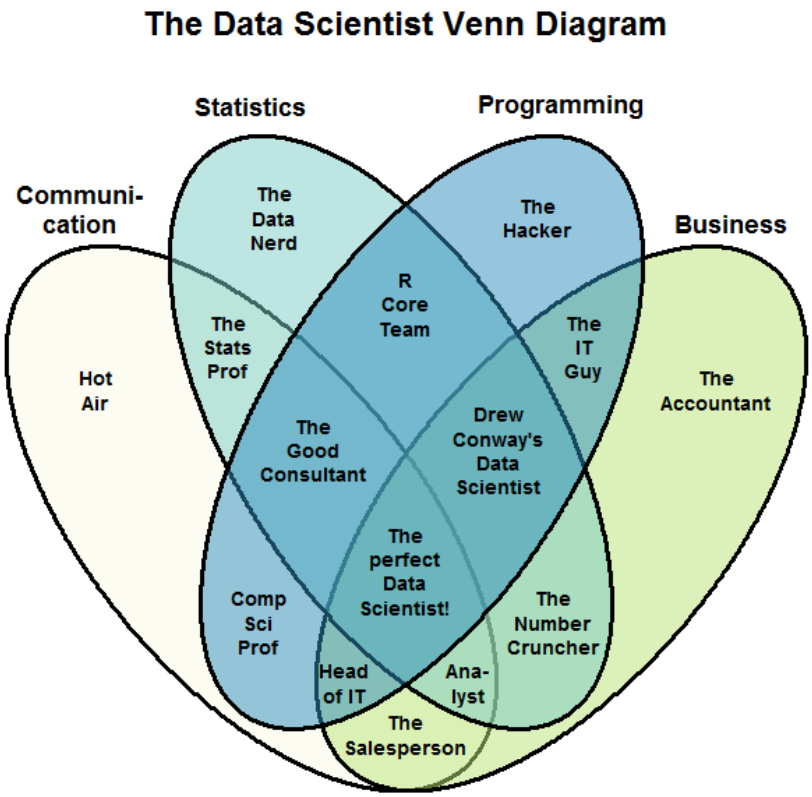
> *„Konvenční statistické metody fungují dobře s malými soubory dat. Dnešní databáze však mohou zahrnovat miliony řádků a desítky sloupců dat. Škálovatelnost je obrovský problém při dolování dat. Další technickou výzvou je vývoj modelů, které dokážou lépe analyzovat data, detekovat nelineární vztahy a interakci mezi prvky.“ (Knowledge at Wharton, 1999)*

Je tedy zřejmé, že pro analýzu dat v novém digitálním světě je zapotřebí rozšířená statistika, což je vyvíjeno v nově vzniklém oboru „datová věda“.

2.3.7 Vymezení obsahu, cílů a procesu datové vědy

Jak bylo zmíněno v minulé podkapitole, datová věda je pojem, který se poprvé použil a uchytil až relativně nedávno. V českém jazyce se o něm i do dnes často referuje (dle originálního anglického názvu) jako o data science.

Záběr oboru data science je dále komplikován častou záměnou s příbuznými obory jako jsou datová analytika a datové inženýrství. Ačkoliv se náplň těchto oborů může navzájem protínat, přece jen se nejedná o jedno a to samé.



Obr. 3: Vennův diagram datového vědce (Castrounis, 2017)

Na vennově diagramu obrázku č. 3 můžeme vidět čtyři disciplíny, které Alex Castrounis ve svém článku „What Is Data Science, and What Does a Data Scientist Do?“ označuje za hlavní pilíře data science.

V ideálním případě by byl datový vědec samozřejmě expert v každém z těchto čtyřech pilířů – realita je však taková, že se většina specializuje spíše na jeden či dva z těchto pilířů. Na základě těchto pilířů by však datový vědec měl být schopen využít

existujících datových zdrojů, a mnohdy i v případě potřeby vytvořit nové, aby z nich mohl vytěžit důležité informace a využitelné poznatky. Toho docílí skrze všemožné statistické metody, využití programovacích jazyků, softwaru apod. Na základě těchto výstupů je pak možné dělat například obchodní rozhodnutí nebo změny pro dosažení obchodních cílů. K tomu je však potřeba data a výsledky jejich analýzy umět adekvátně prezentovat, interpretovat a komunikovat (Castrounis, 2017).

2.3.8 Cíle a výstupy datové vědy

Mezi typické cíle a výstupy datové vědy se dají řadit následující (Castrounis, 2017):

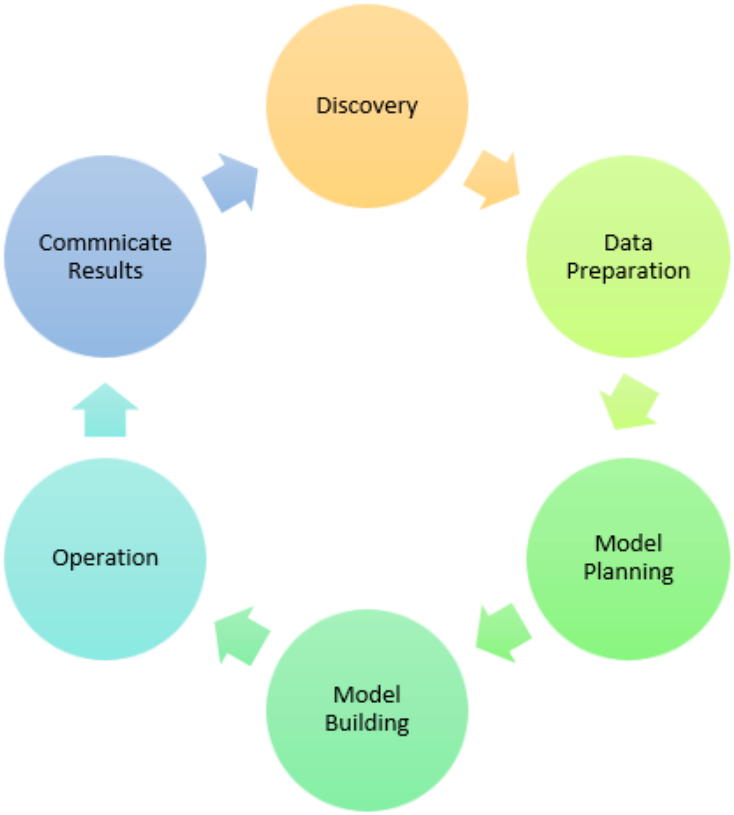
- **Predikce** – predikce výstupu (závislá proměnná) na základě vstupů (nezávislé proměnné). Může se například jednat o predikci výsledků sportovních utkání apod.
- **Klasifikace** – na základě parametrů daný objekt klasifikujeme do kategorií. Například zda email je či není spam.
- **Detekce a seskupování vzorů** – v podstatě se jedná o klasifikaci, ale bez předem známých kategorií. Pokud tedy normální a spam email mají oba svá charakteristická specifika, detekční algoritmus je rozdělí do skupiny A a B, přičemž my je pak pojmenujeme podle vlastní potřeby.
- **Doporučení** – například doporučení obsahu na sociálních sítích, streamovacích platformách typu Netflix, HBO GO apod., nebo doporučení produktů na Amazonu.
- **Rozpoznávání** – např. obrázků, videa, audia, obličeje apod.
- **Využitelné poznatky** – data dashboardy, reporty a další vizualizace.
- **Optimalizace** – například u řízení rizik podniku apod.
- **Segmentace** – využitelné při marketingu pro více demografických skupin apod.
- **Předpovědi** – v obchodním prostředí například předpověď prodeje a tržeb.
- **Řazení** – například kreditní skóre typu FICO apod.

Cílem datové vědy jako takové však není vytváření co nejlepších prediktivních, klasifikačních, optimalizačních, či jiných modelů. Důvodem pro datovou vědu je výzkum a hledání odpovědí na stanovené otázky. Výstupem činnosti datového vědce jsou tedy ve výsledku relevantní informace pro řešení problémů. A z toho důvodu je vennův diagram zobrazený výše natolik různorodý. Datový vědec nesmí být pouze technicky zaměřený člověk, anebo naopak pouze člověk zaměřený na byznys a komunikaci. Musejí se zde protnout všechny zmíněné zaměření, neboť je potřeba mít nejen dobře zodpovězené (kvalitní prediktivní modely), nýbrž i dobře položené otázky (byznys expertíza).

2.3.9 Proces datové vědy

Samotný proces datové vědy je možný roztřídit do šesti základních kroků (Johnson, 2024), které jsou zobrazeny na obrázku č. 4:

- **Sběr dat:** tento krok zahrnuje získání dat, a to ať už z interních či externích zdrojů. Příkladem dat z interních zdrojů může být například dotazník spokojenosti zaměstnanců, z čehož pak HR oddělení může zodpovědět klíčové otázky za účelem vyšší spokojenosti pracovníků. Externí zdroje mohou zahrnovat například data sesbíraná ze sociálních sítí, webserverů, či jiných online zdrojů.
- **Příprava dat:** data, která máme k dispozici, ve většině případů obsahují určité nekonzistence, které je potřeba před samotnou prací s nimi odstranit, abychom se vyvarovali například zkreslení výsledků a podobných problémů. Mezi takové nekonzistence mohou patřit například chybějící hodnoty, prázdná pole, špatný formát sloupců, anebo překlepy – chybějící desetinná čárka nám může například zkreslit průměr hodnot ve sloupci, anebo nekonzistentně používaná velká a malá písmena u stejné kategorie vytvoří jednu či více nežádoucích kategorií navíc. Je tedy klíčové data pročistit a připravit před tím, než se pustíme do práce s nimi.
- **Plánování modelu:** v tomto kroku je potřeba zvolit metody a techniky, pomocí kterých budeme práci s daty provádět. Pro plánování modelu využíváme různé statistické metody a vizualizační nástroje.
- **Budování modelu:** jakmile jsme se rozhodli pro konkrétní model, můžeme jej začít budovat. Dataset, se kterým pracujeme, rozdělíme na trénovací a testovací vzorek (testovací bývá zpravidla výrazně menší – poměr například 80/20). Na trénovací vzorek aplikujeme techniky jako asociace, klasifikace, shlukování apod., pomocí kterých model natrénujeme. V dalším kroku testujeme úspěšnost modelu tím, že jej aplikujeme na testovací vzorek.
- **Dokumentace modelu:** v této fázi dodáváme zhotovený model s kódem a technickou dokumentací. V závislosti na účelu, pro který byl model zhotoven, jej po důkladném testování nasazujeme do produkčního prostředí nebo s ním jednoduše pracujeme pro vlastní potřeby.
- **Komunikace výsledků:** v poslední fázi procesu datové vědy pak interpretujeme a následně prezentujeme výsledky modelu zainteresovaným stranám za účelem například důležitého obchodního rozhodnutí, nasměrování vědeckého výzkumu správným směrem apod



Obr. 4: Proces datové vědy (Johnson, 2024)

2.3.10 „Big data“

Termín „big data“ označuje data primárně na základě jejich velikosti. Mimo velikost však existuje mnoho dalších parametrů, které odlišují „big“ data od „small“. Rob Kitchin a Gavin McArdle (2016) ve svém článku „What makes Big Data, Big Data? Exploring the ontological characteristics of 26 datasets“ uvádějí tři hlavní takové parametry, pomocí kterých můžeme „big data“ charakterizovat:

- **Objem (Volume)** – objemem dat se myslí již zmíněná velikost. Například jen v roce 2014 Facebook musel denně zpracovávat 10 miliard soukromých zpráv, 4.5 miliard stisknutých tlačítek „like“ a 350 milionů nahraných fotek. Sociální sítě jsou tedy jednou doménou, kde se určitě můžeme setkat s „big data“. Dalšími mohou být například bankovní společnosti nebo softwarové společnosti („big tech“) jako například Google (ukládání historie hledání apod.), Apple, Microsoft a podobné.
- **Rychlost (Velocity)** – rychlostí je myšleno převážně tempo, kterým jsou data generována, sbírána a procesována. Díky právě sociálním sítím, ale dalo se říct internetu obecně, jsou data sbírána neuvěřitelnou rychlostí. Z toho důvodu je

potřeba na takovém objemu dat s takovou rychlostí provádět analýzu a zpracovávat je automatizovaně v reálném čase.
- **Různorodost (Variety)** – různorodost se odkazuje na různé formáty a typy dat. Typicky zde charakterizujeme data například podle jejich struktury (viz kapitola „Struktura dat“)

Objem, rychlost a různorodost se běžně označují jako tzv. 3V „big“ dat (odvozeno z anglických názvu volume, velocity a variety). Dále však autoři výše zmíněného článku uvádějí i další, méně často zmiňované parametry:

- **Záběr dat (Exhaustivity)** – tento parametr udává záběr dat ve smyslu obsažení vzorku vs celku. Můžeme mít například data pouze vzorku řekněme padesáti lidí, anebo naopak celé populace daného města či obce.
- **Pružnost a škálovatelnost (Extensionality, scaleability)** – pružnost dat nám udává, jak jednoduše můžeme přidat či změnit objekty v databázi, zatímco škálovatelnost nám udává do jaké do jaké míry můžeme databázi zvětšovat (ve smyslu kolik objektů můžeme přidat).
- **Vztahovost (Relationality)** – vztahovost mezi databázemi znamená, že mají jeden či více společných sloupců, na základě kterých se mohou spojit a tím pádem rozšířit informace o každém dostupném objektu (řádku).

V tabulce č. 2 je znázorněno, jak konkrétní parametry charakterizují „small“ a „big“ data:

Tab. 2: Charakteristika parametrů dat podle jejich rozsahu (Kitchin & McArdle, 2016)

Parametr	„Small Data“	„Big Data“
Objem (Volume)	Limitovaný až velký	Velmi velký
Rychlost (Velocity)	Pomalé, nesouvislé	Rychlé, kontinuální
Různorodost (Variety)	Limitovaná až široká	Široká
Záběr dat (Exhaustivity)	Vzorky	Celé populace
Pružnost a škálovatelnost (Extensionality, scaleability)	Nízká až střední	Silná
Vztahovost (Relationality)	Slabá až silná	Silná

Jak již vyplynulo z definic parametrů, „big“ data primárně charakterizuje jejich objem. Dále jsou však charakterizována vysokou rychlostí a kontinuitou příjmu dat, neboť bez toho by ani nešlo dosáhnout daných rozměrů dat. S tím i přímo souvisí škálovatelnost – databáze, které nejsou přizpůsobené na rapidní rozšiřování a růst dat, nemohou dlouhodobě naplňovat požadavky kladené z definice „big data“. Dále je samozřejmostí sběr obrovského množství různých typů dat, tudíž záběr a různorodost jsou

také klíčovou vlastností. V neposlední řadě je pak velmi neobvyklé, aby byla „big“ data skladována v jedné tabulce – je tedy důležité mít mezi tabulkami v databázi silnou vztahovost, aby se s daty mohlo pracovat napříč celou databází.

2.4 Umělá inteligence a strojové učení

Umělá inteligence (nebo také „AI“ jako zkratka anglického výrazu „artificial intelligence“) se týká vývoje počítačových systémů, které mohou provádět úkoly obvykle vyžadující lidskou inteligenci. Tyto úkoly zahrnují široké spektrum činností, včetně učení, uvažování, řešení problémů, vnímání, porozumění jazyku, anebo i schopnosti sociální interakce. Ve svém jádru se oblast AI snaží vytvářet stroje, které dokážou napodobovat lidské kognitivní funkce, což jim umožňuje autonomně se přizpůsobovat, zlepšovat a provádět úkoly (Copeland, B.J., n.d.).

Základem umělé inteligence je strojové učení – podmnožina AI, která zahrnuje využití algoritmů a statistických modelů, které počítačům umožňují učit se z dat a zlepšovat jejich výkon v průběhu času. Tento proces učení může probíhat prostřednictvím učení pod dohledem, kdy je systém umělé inteligence trénován na označených datech, nebo učení bez dozoru, kdy systém identifikuje vzorce a vztahy v neoznačených datech.

Aplikace umělé inteligence jsou všudypřítomné v našem každodenním životě, od virtuálních osobních asistentů a systémů doporučení až po složité úkoly, jako je lékařská diagnostika a nebo například technologie autonomních vozidel.

2.4.1 Typy umělé inteligence

Umělou inteligenci (AI) lze kategorizovat do různých typů na základě jejích schopností, funkcí a aplikací. Tyto klasifikace poskytují rámec pro pochopení různorodé povahy systémů umělé inteligence. Mezi primární typy umělé inteligence patří úzká (slabá) AI, obecná (silná) AI a umělá superinteligence (Joshi, 2019).

Úzká umělá inteligence, známá také jako slabá umělá inteligence, označuje systémy umělé inteligence navržené pro konkrétní úkol nebo omezený rozsah úkolů. Tyto systémy vynikají ve své předdefinované doméně, ale postrádají všestrannost a adaptabilitu, kterou lze nalézt v lidské inteligenci. Příklady úzké umělé inteligence zahrnují virtuální osobní asistenty jako Siri nebo Alexa, software pro rozpoznávání obrazu a řeči a algoritmy doporučení používané streamovacími platformami. Přes jejich cílené aplikace se úzké systémy AI ukázaly jako vysoce efektivní a staly se nedílnou součástí našeho každodenního života.

Naproti tomu obecná (silná) AI (anglicky „artificial general intelligence“, nebo také AGI) si klade za cíl replikovat široké kognitivní schopnosti lidské inteligence. Takové systémy by měly schopnost porozumět, učit se a vykonávat jakýkoli intelektuální úkol, který dokáže lidská bytost. Dosažení obecné umělé inteligence zůstává obrovskou výzvou, protože vyžaduje, aby stroje projevovaly porozumění v různých

oblastech a prokazovaly úroveň adaptability a řešení problémů podobnou lidskému poznání.

Umělá superinteligence (ASI) je teoretický koncept, který předpokládá, že AI překoná lidskou inteligenci ve všech oblastech. Tato úroveň inteligence by potenciálně umožnila strojům překonat nejchytřejší lidské mysli ve všech myslitelných intelektuálních snahách. Vývoj umělé superinteligence vyvolává hluboké etické a existenční otázky, které vyžadují pečlivé zvážení a etické pokyny k zajištění odpovědného nasazení a kontroly.

Další kategorizace umělé inteligence zahrnují reaktivní stroje, omezenou paměť, teorii mysli a umělou inteligenci s vědomím. Reaktivní stroje fungují na základě předem definovaných pravidel bez učení nebo přizpůsobování. Systémy s omezenou pamětí se na druhou stranu mohou učit z historických dat, díky čemuž jsou přizpůsobivější. Teorie mysli odkazuje na systémy umělé inteligence, které rozumí lidským emocím, přesvědčením a záměrům, zatímco umělá inteligence s vědomím sebe sama považuje za živou a sebe uvědomělou bytost, což je koncept, který zůstává spekulativní a dalekosáhlý (Hintze, 2016).

2.4.2 Strojové učení a jeho typy

Strojové učení (ML) lze definovat jako obor, který umožňuje počítačům dělat přesné predikce pomocí minulých zkušeností. Toho je dosaženo využitím algoritmů a technik, které analyzují data a extrahují z nich vzory nebo vztahy. Strojové učení zažilo významný vývoj, především díky pokrokům v kapacitě úložiště a výkonu zpracování počítačů (Baştanlar & Özuysal, 2013).

Strojové učení se stalo základem moderní analýzy dat a počítačových technologií, umožňující inteligentní funkcionality napříč různými aplikacemi. Algoritmy ML usnadňují učení a zdokonalování zkušenostmi bez explicitního programování, což je nedocenitelné v době hojnosti dat a automatizace. Tyto algoritmy lze obecně rozdělit do čtyř hlavních typů – učení s učitelem, učení bez učitele, polo-supervizované učení a učení metodou zpětné vazby (Sarker, 2021c):

- **Učení s dohledem (supervizované):** v učení s dohledem je algoritmus trénován na označeném datasetu, kde je každý datový bod párován s odpovídajícím štítkem nebo výstupem. Algoritmus se učí mapovat vstupy na výstupy na základě příkladů vstup-výstup. Běžné úkoly v učení s učitelem zahrnují klasifikaci a regresi. Klasifikace zahrnuje předpovídání diskrétních třídních štítků, zatímco regrese předpovídá spojité číselné hodnoty.
- **Učení bez dohledu (nesupervizované):** učení bez dohledu pracuje s neoznačenými daty a má za cíl odhalit skryté vzory nebo struktury v datasetu. Algoritmus identifikuje vnitřní vztahy nebo shluky mezi datovými body bez explicitního vedení štítky. Shlukování a redukce dimenzionality jsou typické úkoly v učení bez učitele. Shlukování rozděluje data do skupin se podobnými charakteristikami,

zatímco techniky redukce dimenzionality si klade za cíl komprimovat data zachováním jejich podstatných vlastností.

- **Polo-supervizované učení:** Polo-supervizované učení kombinuje prvky jak učení s dohledem, tak učení bez dohledu. Využívá malé množství označených dat vedle většího množství neoznačených dat ke zlepšení přesnosti učení. Integrováním označených a neoznačených dat polo-supervizované učení řeší výzvy spojené s omezenou dostupností označených dat a zároveň těží z škálovatelnosti nesupervizovaných přístupů.
- **Učení metodou zpětné vazby:** Učení metodou zpětné vazby zahrnuje agenta, který se učí provádět postupné rozhodnutí interakcí s prostředím. Agent dostává zpětnou vazbu ve formě odměn nebo trestů na základě svých akcí, které řídí jeho učící se proces. Prostřednictvím pokusů a omylů se agent učí optimální strategie k maximalizaci kumulativních odměn v průběhu času. Učení metodou zpětné vazby je běžně používáno v situacích, které vyžadují rozhodování v nejistotě, jako jsou hraní her, robotika a řízení autonomních vozidel.

2.4.2.1 Lineární a ne-lineární modely strojového učení

V oblasti strojového učení leží jeden základní koncept v odlišení linearity a nelinearity. Lineárnost označuje vlastnost systému nebo modelu, kde výstup je přímo úměrný vstupu, zatímco nelinearita naznačuje, že vztah mezi vstupem a výstupem je složitější a nedá se vyjádřit jako jednoduchá lineární funkce.

Lineární modely často představují jednoduchý a efektivní přístup v mnoha případech. V podstatě lineární model přizpůsobí přímou linii datům, což umožňuje predikce na základě lineárního vztahu mezi vstupními prvky a výstupní proměnnou.

Nelineární modely se projevují v různých podobách, od polynomiálních modelů, které se přizpůsobují křivkám dat, po neuronové sítě schopné rozluštit složité vzory ve vysoko-dimenzionálních datech. Tyto modely překračují lineární v odhalování složitých vztahů mezi proměnnými, což je činí mocnějšími, zejména v klasifikačních úkolech, kde převládají nelineární hranice rozhodování.

Volba mezi lineárními a nelineárními modely závisí na konkrétní povaze problému. Lineární modely postačují pro jednodušší problémy, ale mohou selhat v řešení složitějších úkolů. Naopak nelineární modely nabízejí větší sílu, ale přinášejí větší komplexitu a nároky na zdroje (Zarra, R., 2023).

2.4.2.2 Nejpoužívanější modely strojového učení

Jedním z nejpoužívanějších lineárních modelů strojového učení je regrese, a to konkrétně lineární a logistická. Regrese obecně se snaží hledat vztahy mezi nezávislými a závislými proměnnými za účelem predikce určitého výstupu. Lineární regrese se snaží predikovat výsledek proměnné s kontinuální hodnotou – v praxi se například používá pro predikci cen akcií, počasí nebo poptávky zboží. Logistická regrese se naopak

používá při binární výstupu. Měří tedy pravděpodobnost výstupu od 0 % do 100 %, na základě čehož predikuje buď výstup 0, anebo 1 (Lawton, G., 2023).

Naopak jedním z nejpoužívanějších ne-lineárních modelů je tzv. „decision tree“, neboli rozhodovací strom. Jedná se o schéma složené z mnoha rozhodovacích uzlů, kde každý z nich představuje určitou otázku, s jejíž odpovědí se algoritmus posune na další rozhodovací uzel. Na konci tohoto procesu algoritmus dorazí k odpovědi na cílovou otázku – např. „vyhraje hráč 1?“. Nicméně jednoduchý model rozhodovacího stromu může být zaujatý a vést k přeučení[1]. Tyto problémy můžeme obcházet například spojením několika rozhodovacích stromů dohromady – jedním takovým modelem je například „random forest“ (IBM, n.d.).

2.4.3 Hluboké učení a neuronové sítě

Hluboké učení je podmnožinou strojového učení a umělé inteligence, která využívá neuronové sítě s více vrstvami k reprezentaci a učení vzorů z dat. Hluboké učení exceluje v učení reprezentací dat automatizovaným způsobem, což ho činí cenným pro různé aplikace, jako jsou klasifikační a regresní úkoly. Liší se od standardních metod strojového učení svou efektivitou s narůstajícím objemem dat, ačkoliv obvykle vyžaduje delší dobu trénování kvůli velkému množství parametrů zapojených do procesu (Sarker, 2021b).

Neuronová síť je pak výpočetní model inspirovaný strukturou a fungováním lidského mozku. Skládá se z propojených uzlů nazývaných neurony uspořádaných do vrstev. Každý neuron přijímá vstupní signály, zpracovává je pomocí vah a zkreslení a aplikuje aktivační funkci k produkci výstupního signálu. Spojení mezi neurony jsou reprezentována váhami, které se během trénování upravují k optimalizaci výkonu sítě na daném úkolu. Neuronové sítě jsou schopny učit se složité vzory z dat a jsou široce používány v různých oblastech pro úkoly jako klasifikace, regrese a rozpoznávání vzorů (Sarker, 2021b).

2.4.3.1 Aktivační a ztrátové funkce neuronové sítě

V neuronové síti se na vážený součet vstupů aplikuje aktivační funkce k určení výstupu neuronu. Zavádí nelinearitu do sítě, což jí umožňuje učit se složité vzory a vztahy v datech. Běžné aktivační funkce zahrnují sigmoidu, tanh, ReLU (Rectified Linear Unit) a softmax, každá s vlastními charakteristikami a použitím.

Ztrátová funkce, také nazývaná nákladová nebo chybová funkce, měří rozdíl mezi předpovězeným výstupem neuronové sítě a skutečným cílovým výstupem pro daný vstup. Kvantifikuje, jak dobře síť pracuje na konkrétním úkolu během trénování. Cílem trénování je minimalizovat ztrátovou funkci, což je obvykle dosaženo pomocí

[1] Přeučení je nežádoucí jev ve strojovém učení, při kterém model zvládne poskytnou přesné předpovědi na trénovacím vzorku dat, ale ne pro nová data (Amazon Web Services, n.d.).

optimalizačních algoritmů jako je gradientový sestup. Různé úkoly mohou vyžadovat různé typy ztrátových funkcí; například střední kvadratická chyba (MSE) je často používána pro regresní úkoly, zatímco ztrátová funkce cross-entropy je běžně používána pro klasifikační úkoly (Sarker, 2021b).

3 Cíle práce a výzkumné otázky

Cílem práce je vytvoření a optimalizace prediktivních modelů strojového učení natrénovaných na herních charakteristikách tenisových utkání a následná analýza ziskovosti konkrétních sázkových strategií pomocí vygenerovaného modelu s nejvyšší prediktivní přesností. Pro splnění stanovených cílů byly zvoleny následující výzkumné otázky:

1. Který z vybraných modelů strojového učení je na predikci výsledků tenisových zápasů nejpřesnější?
2. Jaké proměnné mají největší prediktivní váhu u nejpřesnějšího modelu?
3. Je využití metod umělé inteligence pro predikci výsledků tenisových zápasů přesnější než kurzy vypsané bookmakery?
4. Jaký je rozdíl ve výnosnosti retrospektivního sportovního sázení s využitím strojového učení při komparaci se sázením podle kurzů od bookmakerů?

4 Metodika

Tato kvantitativní, prediktivně komparativní studie má za cíl zpracovat surová data a vytvořit prediktivní modely strojového učení pro retrospektivní predikci výsledků tenisových zápasů a následnou analýzu ziskovosti sportovního sázení při využití vytvořených predikcí. V této kapitole je popsána veškerá metodika, která byla k dosažení těchto cílů použita.

4.1 Charakteristika dat

Originální data byla získána z webu *Kaggle.com* (Cantagallo, E., n.d.). Charakteristika dat je uvedena v tabulce č. 3.

Tab. 3: Data v originální podobě před manipulací

Název sloupce	Popis	Hodnota	Typ dat
ATP	Číslo turnaje	Celé číslo	Kvalitativní, nominální
Location	Místo konání turnaje	Text	Kvalitativní, nominální
Tournament	Jméno turnaje	Text	Kvalitativní, nominální
Date	Datum konání zápasu	Text	Kvantitativní, intervalová
Series	Jméno ATP série	Text	Kvalitativní, nominální
Court	Typ kurtu	Text	Kvalitativní, binární
Surface	Povrch kurtu	Text	Kvalitativní, nominální
Round	Kolo zápasu	Text	Kvalitativní, ordinální
Best of	Maximální možný počet setů v zápase	Celé číslo	Kvalitativní, binární
Winner	Jméno vítěze	Text	Kvalitativní, nominální
Loser	Jméno poraženého	Text	Kvalitativní, nominální
Wrank	ATP žebříček vítěze	Celé číslo	Kvantitativní, ordinální
Lrank	ATP žebříček poraženého	Celé číslo	Kvantitativní, ordinální
Wpts	ATP body vítěze	Celé číslo	Kvantitativní, poměrová
Lpts	ATP body poraženého	Celé číslo	Kvantitativní, poměrová
W1	Počet vyhraných gemů vítězem v prvním setu	Celé číslo	Kvantitativní, ordinální
L1	Počet vyhraných gemů poraženým v prvním setu	Celé číslo	Kvantitativní, ordinální

Název sloupce	Popis	Hodnota	Typ dat
W2	Počet vyhraných gemů vítězem ve druhém setu	Celé číslo	Kvantitativní, ordinální
L2	Počet vyhraných gemů poraženým ve druhém setu	Celé číslo	Kvantitativní, ordinální
W3	Počet vyhraných gemů vítězem ve třetím setu	Celé číslo	Kvantitativní, ordinální
L3	Počet vyhraných gemů poraženým ve třetím setu	Celé číslo	Kvantitativní, ordinální
W4	Počet vyhraných gemů vítězem ve čtvrtém setu	Celé číslo	Kvantitativní, ordinální
L4	Počet vyhraných gemů poraženým ve čtvrtém setu	Celé číslo	Kvantitativní, ordinální
W5	Počet vyhraných gemů vítězem v pátém setu	Celé číslo	Kvantitativní, ordinální
L5	Počet vyhraných gemů poraženým v pátém setu	Celé číslo	Kvantitativní, ordinální
Wsets	Počet vyhraných setů vítězem	Celé číslo	Kvantitativní, ordinální
Lsets	Počet vyhraných setů poraženým	Celé číslo	Kvantitativní, ordinální
Comment	Poznámka k zápasu	Text	Kvalitativní, nominální
B365W	Bet365 kurz vítěze	Desetinné číslo	Kvantitativní, poměrová
B365L	Bet365 kurz poraženého	Desetinné číslo	Kvantitativní, poměrová
PSW	Pinnacles Sports kurz vítěze	Desetinné číslo	Kvantitativní, poměrová
PSL	Pinnacles Sports kurz poraženého	Desetinné číslo	Kvantitativní, poměrová
MaxW	Max kurz vítěze	Desetinné číslo	Kvantitativní, poměrová
MaxL	Max kurz poraženého	Desetinné číslo	Kvantitativní, poměrová
AvgW	Průměrný kurz vítěze	Desetinné číslo	Kvantitativní, poměrová

Název sloupce	Popis	Hodnota	Typ dat
AvgL	Průměrný kurz poraženého	Desetinné číslo	Kvantitativní, poměrová
EXW	Expekt kurz vítěze	Desetinné číslo	Kvantitativní, poměrová
EXL	Expekt kurz poraženého	Desetinné číslo	Kvantitativní, poměrová
LBW	Ladbrokes kurz vítěze	Desetinné číslo	Kvantitativní, poměrová
LBL	Ladbrokes kurz poraženého	Desetinné číslo	Kvantitativní, poměrová
SJW	Stan James kurz vítěze	Desetinné číslo	Kvantitativní, poměrová
SJL	Stan James kurz poraženého	Desetinné číslo	Kvantitativní, poměrová
UBW	Unibet kurz vítěze	Desetinné číslo	Kvantitativní, poměrová
UBL	Unibet kurz poraženého	Desetinné číslo	Kvantitativní, poměrová
pl1_flag	Národnost vítěze	Text	Kvalitativní, nominální
pl1_year_pro	Rok zahájení profesionální kariéry vítěze	Celé číslo	Kvalitativní, nominální
pl1_weight	Váha vítěze	Celé číslo	Kvantitativní, poměrová
pl1_height	Výška vítěze	Celé číslo	Kvantitativní, poměrová
pl1_hand	Dominantní ruka vítěze	Text	Kvalitativní, binární
pl2_flag	Národnost poraženého	Text	Kvalitativní, nominální
pl2_year_pro	Rok zahájení profesionální kariéry poraženého	Celé číslo	Kvalitativní, nominální
pl2_weight	Váha poraženého	Celé číslo	Kvantitativní, poměrová
pl2_height	Výška poraženého	Celé číslo	Kvantitativní, poměrová
pl2_hand	Dominantní ruka poraženého	Text	Kvalitativní, binární

4.2 Manipulace s daty

Pro manipulaci s daty byl využit programovací jazyk *Python* a knihovna pro datovou analýzu *Pandas*. Cílem manipulace s daty bylo data pro naši práci důkladně vyčistit, vytvořit či přetransformovat vhodné nezávislé proměnné (prediktory) a definovat závislou proměnnou pro následné trénování modelů strojového učení. Podrobný postup

s každým jednotlivým krokem čištění a manipulace s daty a všechny použité funkce a metody jsou popsány v příloze A.

4.3 Tvorba a trénování modelů strojového učení

Modely strojového učení pro predikci výsledků tenisových zápasů byly zvoleny dohromady tři – random forest, logistická regrese a sekvenční neuronová síť.

4.3.1 Random forest

Pro vytvoření modelu random forest a jeho trénování na datech byl použit programovací jazyk *Python*, knihovna pro datovou analýzu *Pandas* a knihovna pro strojové učení *scikit-learn*.

Bylo potřeba definovat prediktory a závislou proměnnou, rozdělit data na trénovací a testovací vzorek podle data konání zápasů. Následně bylo využito třídy *StandardScaler* z knihovny *scikit-learn* pro standardizaci dat jak v trénovacím, tak v testovacím vzorku. Dále byl zvolen ze stejné knihovny klasifikátor *RandomForestClassifier*, který byl natrénován na trénovacím vzorku a následně aplikován na vzorek testovací. Podrobnější postup a veškerý použitý kód se nachází v příloze B.2.

4.3.2 Logistická regrese

Pro vytvoření modelu logistické regrese a jeho trénování na datech byl použit programovací jazyk *Python*, knihovna pro datovou analýzu *Pandas*, knihovna pro strojové učení *scikit-learn* a statistická knihovna *statsmodels*.

Bylo potřeba definovat prediktory a závislou proměnnou, rozdělit data na trénovací a testovací vzorek podle data konání zápasů. Následně bylo využito třídy *StandardScaler* z knihovny *scikit-learn* pro standardizaci dat jak v trénovacím, tak v testovacím vzorku. Dále byla s pomocí knihovny *statsmodels* do trénovacího vzorku přidána konstanta a vytvořen model logistické regrese pomocí třídy *Logit* (opět z knihovny statsmodels), na kterém byl natrénován trénovací vzorek. Podrobnější postup a veškerý použitý kód se nachází v příloze B.3.

4.3.3 Sekvenční neuronová síť

Pro vytvoření modelu sekvenční neuronové sítě a jeho trénování na datech byl použit programovací jazyk *Python*, knihovna pro datovou analýzu *Pandas*, knihovna pro strojové učení *scikit-learn* a knihovna pro deep learning *TensorFlow*.

Bylo potřeba definovat prediktory a závislou proměnnou, rozdělit data na trénovací a testovací vzorek podle data konání zápasů. Následně bylo využito třídy *StandardScaler* z knihovny *scikit-learn* pro standardizaci dat jak v trénovacím, tak v testovacím vzorku. Dále byl s pomocí knihovny *TensorFlow* vytvořen model sekvenční

neuronové sítě s pěti *Dense* vrstvami. Pro 1., 2., 4. a 5. skrytou vrstvu byla jako aktivační funkce použita *Sigmoid*; pro 3. vrstvu byla použita aktivační funkce *ReLU*. Byly zároveň vyzkoušeny i ostatní varianty architektury neuronové sítě (jiné aktivační funkce, v jiném pořadí apod.), nicméně tato architektura byla v průměru nejúspěšnější, alespoň co se přesnosti predikcí týče.

Při kompilaci modelu byl jako optimalizační algoritmus použit *Adam* a jako ztrátová funkce byla zvolena *Binary Cross Entropy*. Podrobnější postup a veškerý použitý kód se nachází v příloze B.4.

4.3.4 Měření úspěšnosti jednotlivých modelů

Jako sjednocený výstup všech třech modelů strojového učení byla použita confusion matice. Hlavní informací je však pro tuto studii přesnost každého modelu – tedy v kolika případech se model svou predikcí trefil do reálného výsledku zápasu.

Ostatní parametry confusion matice (senzitivita, specificita, pozitivní a negativní prediktivní hodnota) mohou být potenciálně hodnotné, nicméně jelikož je výsledkem naší predikce hráč 1 nebo hráč 2 (a tudíž nikoliv obvyklé *ano* a *ne*), nemají pro nás tyto parametry tak hlubokou informační hodnotu.

V případě modelů random forest a logistické regrese bylo také vyhodnoceno, které nezávislé proměnné mají nejvyšší statistickou významnost (u modelu sekvenční neuronové sítě tato informace není dostupná).

4.4 Volba a algoritmizace sázkových strategií

Pro sázkové strategie byly zvoleny dva základní přístupy – sázení s fixní částkou a sázení s negativní progresí. U fixní částky sázíme vždy stejně vysokou částkou bez ohledu na výsledky našeho sázení. U sázení s negativní progresí, a to konkrétně podle systému *Martingale*, vždy dvojnásobíme naši další sázku po každé prohře, přičemž po každé výhře se další sázka vrací na její původní zvolenou výši.

Další distinkce mezi strategiemi jsou sázení podle vypsaných kurzů a sázení podle vytvořených predikcí. Dále pak sázíme buď na všechny zápasy v testovacím vzorku, anebo pouze na zápasy, které splňují námi stanovené pravděpodobnostní a kurzové podmínky.

V neposlední řadě je pak nutné doplnit, že před použitím každé sázkové strategie jsou řádky datového setu s tenisovými zápasy náhodně promíchány. Na strategie se sázením s fixní částkou promíchání zápasů nemá vliv, ale u sázení s Martingale systémem to při každém pokusu může výrazně změnit výsledek a lépe to tak simuluje možné budoucí zápasy. U všech strategií s *Martingale* systémem je z tohoto důvodu výsledek průměrem 100 pokusů dané strategie. Kód pro každou ze strategií se nachází v příloze C.

4.4.1 Přehled zvolených sázkových strategií

Zvoleno bylo celkem 9 sázkových strategií. Označení a popis všech strategií je uveden v následující tabulce č. 4.

Tab. 4: Zvolené sázkové strategie

Označení	Popis strategie
A	Sázení na všechny zápasy fixní částkou podle kurzů od bookmakerů
B	Sázení na všechny zápasy fixní částkou podle predikcí LR
C	Sázení na vybrané zápasy fixní částkou podle kurzů od bookmakerů
D1	Sázení na vybrané zápasy fixní částkou podle predikcí LR (bez spodní hranice vypsaného kurzu)
D2	Sázení na vybrané zápasy fixní částkou podle predikcí LR (se spodní hranicí vypsaného kurzu)
E	Sázení na všechny zápasy s negativní progresí podle kurzů od bookmakerů
F	Sázení na všechny zápasy s negativní progresí podle predikcí LR
G	Sázení na vybrané zápasy s negativní progresí podle kurzů od bookmakerů
H	Sázení na vybrané zápasy s negativní progresí podle predikcí LR

4.4.2 Podrobný popis zvolených sázkových strategií

4.4.2.1 Výběr hráče

U strategií A, C, E a G sázíme na hráče podle vypsaných kurzů od bookmakerů – vždy sázka na hráče, u kterého je vypsaný nižší kurz (tj. předpokládaná výhra).

U strategií B, D1, D2, F a H sázíme na hráče podle predikcí modelu logistické regrese. Pokud je predikce vyšší než 0,5 (50 % šance na výsledek 1, tedy výhru hráče 1), předpokládáme výhru hráče 1, na kterého následně sázíme. V opačném případě (predikce < 0,5) se sází na hráče 2.

4.4.2.2 Volba výše sázky

U strategií A, B, C, D1 a D2 sázíme vždy zvolenou fixní částkou bez ohledu na výsledek předchozí sázky – pokud zvolíme jednotku 100, pak sázíme na každý zápas přesně 100.

U strategií E, F, G a H aplikujeme negativní progresi podle *Martingale* systému – výši sázky tedy upravujeme v závislosti na výsledku předchozí sázky (při prohře dvojnásobek předchozí sázky, při výhře se další sázka srazí zpět na původní zvolenou výši).

4.4.2.3 Filtrace zápasů

Sázení na všechny zápasy

V případě strategií A, B, E a F se sází na všechny zápasy.

Sázení na vybrané zápasy

- U **strategie C** sázíme pouze v případě, kdy je na předpokládaného výherce vypsaný kurz 1,43 a níže (odpovídá cca pravděpodobnosti na výhru 70 %). Jinými slovy – vsadíme si pouze v tom případě, kdy předpokládáme 70 % šanci na výhru.
- U **strategie D1** sázíme pouze v případě, kdy je na předpokládaného výherce predikovaná pravděpodobnost výhry 70 % a více podle modelu logistické regrese.
- U **strategie D2** se přebírá stejná podmínka, která je u strategie D1, ale navíc je přidána podmínka, že sázíme pouze na ty zápasy, kde je u předpokládaného výherce vypsaný kurz 1,8 a více. Musí být tedy pravdivé obě podmínky – 70 % a více pravděpodobnost výhry a 1,8 a více vypsaný kurz.
- U **strategie G** se snažíme simulovat 50/50 šanci na výhru, pro kterou byl *Martingale* systém původně vytvořen. Na předpokládaného výherce tedy vsadíme pouze v případě, kdy rozdíl kurzu na hráče 1 a hráče 2 nabývá absolutní hodnoty 0,2 – pokud je tedy kurz na předpokládaného výherce například 1.5, vsadíme si pouze tehdy, pokud je na druhého hráče vypsán kurz v rozmezí 1,3–1,7.
- U **strategie H** opět simulujeme 50/50 šanci na výhru. Nicméně v této strategii využíváme predikovanou pravděpodobnost výhry pomocí logistické regrese – pokud je predikovaná pravděpodobnost v rozmezí 48–52 %, tak si na předpokládaného výherce vsadíme.

4.4.3 Vyhodnocení úspěšnosti sázkových strategií

Primárním zvoleným ukazatelem úspěšnosti daných sázkových strategií je výnosnost sázení, kterou získáme vydělením čistého zisku celkovou sumou vynaložených nákladů:

$$Výnosnost = \frac{Čistý\ zisk}{Suma\ všech\ sázek}$$

Čistým ziskem je myšlena suma výsledků všech sázek. Při sázce částkou 500 na kurz 1,5 je výsledek prohry -500 a výsledek výhry 250. Sumou všech sázek je myšlen počet sázek vynásobený fixní částkou – při fixní sázce částkou 500 na 10 zápasů tedy suma všech sázek činí 5000.

5 Výsledky

5.1 Výstup manipulace s daty

Nezávislé proměnné (prediktory), které jsou výsledkem čištění a manipulace s daty, a které budou použity pro trénování modelů strojového učení, jsou uvedeny v následující tabulce č. 5.

Tab. 5: Nezávislé proměnné po manipulaci s daty

Název	Popis	Hodnota	Typ dat
pl1_rank	ATP umístění hráče 1	Celé číslo	Kvantitativní, ordinální
pl2_rank	ATP umístění hráče 2	Celé číslo	Kvantitativní, ordinální
pl1_pts	ATP body hráče 1	Celé číslo	Kvantitativní, poměrová
pl2_pts	ATP body hráče 2	Celé číslo	Kvantitativní, poměrová
pl1_year_pro	rok zahájení profesionální kariéry hráče 1	Celé číslo	Kvalitativní, nominální
pl2_year_pro	rok zahájení profesionální kariéry hráče 2	Celé číslo	Kvalitativní, nominální
pl1_height	tělesná výška hráče 1	Celé číslo	Kvantitativní, poměrová
pl2_height	tělesná výška hráče 2	Celé číslo	Kvantitativní, poměrová
pl1_weight	tělesná váha hráče 1	Celé číslo	Kvantitativní, poměrová
pl2_weight	tělesná váha hráče 2	Celé číslo	Kvantitativní, poměrová
pl1_avg_bookmaker_odds	průměr vypsaných kurzů na hráče 1	Desetinné číslo	Kvantitativní, poměrová
pl2_avg_bookmaker_odds	průměr vypsaných kurzů na hráče 2	Desetinné číslo	Kvantitativní, poměrová
pl1_hand_code	dominantní ruka hráče 1	Celé číslo	Kvalitativní, binární
pl2_hand_code	dominantní ruka hráče 2	Celé číslo	Kvalitativní, binární
pl1_total_games	celkový počet her v kariéře hráče 1	Celé číslo	Kvantitativní, poměrová

Název	Popis	Hodnota	Typ dat
pl2_total_games	celkový počet her v kariéře hráče 2	Celé číslo	Kvantitativní, poměrová
pl1_win_rate	míra výher v kariéře hráče 1	Desetinné číslo	Kvantitativní, poměrová
pl2_win_rate	míra výher v kariéře hráče 2	Desetinné číslo	Kvantitativní, poměrová
pl1_swrate	míra výher na konkrétním povrchu kurtu v kariéře hráče 1	Desetinné číslo	Kvantitativní, poměrová
pl2_swrate	míra výher na konkrétním povrchu kurtu v kariéře hráče 2	Desetinné číslo	Kvantitativní, poměrová
rank_diff	rozdíl ATP umístění hráče 1 a hráče 2	Celé číslo	Kvantitativní, poměrová
pts_diff	rozdíl ATP bodů hráče 1 a hráče 2	Celé číslo	Kvantitativní, poměrová
wrate_diff	rozdíl míry výher v kariéře hráče 1 a hráče 2	Desetinné číslo	Kvantitativní, poměrová
swrate_diff	rozdíl míry výher na konkrétním povrchu kurtu v kariéře hráče 1 a hráče 2	Desetinné číslo	Kvantitativní, poměrová
series_code	ATP série	Celé číslo	Kvalitativní, nominální
court_code	typ kurtu (vnitřní/venkovní)	Celé číslo	Kvalitativní, binární
surface_code	povrch kurtu	Celé číslo	Kvalitativní, nominální
round_code	kolo tenisového zápasu	Celé číslo	Kvalitativní, ordinální

Za závislou proměnnou pro trénování a testování modelů strojového učení pak byl zvolen výsledek zápasu. Jedná se konkrétně o hodnotu *result*, která nabývá hodnot 0 (výhra hráče 2) a 1 (výhra hráče 1). Dá se tedy označit za kvalitativní, binární datový typ

5.2 Výstupy vybraných modelů strojového učení a úspěšnost vypsaných kurzů

5.2.1 Random forest

Podle tabulky č. 6 mají u modelu random forest nejvyšší významnost (> 0,05) prediktory pl2_avg_bookmaker_odds, pl1_avg_bookmaker_odds, pl2_swrate, swrate_diff, pl1_swrate.

Tab. 6: Významnost atributů u výstupu random forest

Atribut (Nezávislá proměnná)[2]	Významnost atributu
pl2_avg_bookmaker_odds	0,1137
pl1_avg_bookmaker_odds	0,1015
pl2_swrate	0,0550
swrate_diff	0,0530
pl1_swrate	0,0504
wrate_diff	0,0448
pl1_pts	0,0435
pts_diff	0,0430
pl1_win_rate	0,0429
pl2_pts	0,0416
pl2_win_rate	0,0398
pl1_rank	0,0394
pl2_rank	0,0391
rank_diff	0,0357
pl2_total_games	0,0334
pl1_total_games	0,0330
pl2_weight	0,0244
pl1_weight	0,0243
pl2_year_pro	0,0241
pl1_year_pro	0,0241
pl2_height	0,0202
pl1_height	0,0198
round_code	0,0144
series_code	0,0142
surface_code	0,0075
court_code	0,0051
best_of	0,0043

[2] Charakteristika jednotlivých atributů je uvedena v tabulce č. 5 v kapitole 5.1.

Atribut (Nezávislá proměnná)[2]	Významnost atributu
pl1_hand_code	0,0040
pl2_hand_code	0,0037

Celková přesnost (anglicky accuracy) prediktivního modelu random forest je 67,04 %. Senzitivita (anglicky true positive rate) pro hodnotu 0 je 66,38 %; specificita (anglicky true negative rate) pro výsledek 1 je 67,76 %. Pozitivní prediktivní hodnota (anglicky zkratka PPV) modelu pro hodnotu 0 je 69,11 %; negativní prediktivní hodnota (anglicky zkratka PV) pro výsledek 1 je 64,98 % (viz tabulka č. 7).

Tab. 7: Confusion matice pro random forest

Výstup \ Cíl	Hráč 2	Hráč 1	Celkem
Hráč 2	**2172** 34,56 %	**971** 15,45 %	**3143** 69,11 % 30,89 %
Hráč 1	**1100** 17,50 %	**2041** 32,48 %	**3141** 64,98 % 35,02 %
Celkem	**3272** 66,38 % 33,62 %	**3012** 67,76 % 32,24 %	**4249 / 6284** 67,04 % 32,96 %

Poznámka: Confusion matice zaznamenává shody (zelená barva) a neshody (červená barva) výstupu modelu strojového učení (tj. predikce výsledku zápasu) a cíle (tj. reálného výsledku zápasu).

5.2.2 Logistická regrese

V tabulce č. 8 jsou uvedeny základní informace k výsledku modelu logistické regrese. Důležitým ukazatelem schopnosti nezávislých proměnných vysvětlit variabilitu závislé proměnné je tzv. pseudo R-kvadrát, který je v našem případě 19,5 %.

Tab. 8: Obecný výsledek logistické regrese

Závislá proměnná	result
Model	Logit
Metoda	MLE
Konvergace	True
Typ kovariance	non-robust
Počet observací	26.404
Stupeň volnosti zbytků	26.374
Stupeň volnosti modelu	29

Pseudo R-kvadrát	0,1953
Log-Likelihood	-14.728
LL-Null	-18.302
LLR p-hodnota	0,000

Nejvyšší statistickou významnost (p-hodnota < 0,05) prediktory pl2_rank, pl1_avg_bookmaker_odds, pl2_avg_bookmaker_odds, pl2_win_rate, pl1_swrate, pl2_swrate (viz tabulka č. 9).

Tab. 9: P-hodnoty nezávislých proměnných u logistické regrese

Nezávislá proměnná[3]	p-hodnota
pl2_rank	0,000
pl1_avg_bookmaker_odds	0,000
pl2_avg_bookmaker_odds	0,000
pl2_win_rate	0,000
pl1_swrate	0,000
pl2_swrate	0,000
rank_diff	0,075
pl1_win_rate	0,100
wrate_diff	0,146
pl2_height	0,216
swrate_diff	0,240
pl2_weight	0,261
pts_diff	0,320
court_code	0,345
pl1_hand_code	0,513
pl2_year_pro	0,529
pl2_total_games	0,557
round_code	0,678
pl1_rank	0,695
surface_code	0,709
best_of	0,717
pl1_pts	0,769
pl1_height	0,774
pl2_pts	0,779
series_code	0,834
const	0,858
pl1_total_games	0,862

[3] Charakteristika jednotlivých nezávislých proměnných je uvedena v tabulce č. 5 v kapitole 5.1.

Nezávislá proměnná[3]	p-hodnota
pl2_hand_code	0,960
pl1_year_pro	0,976
pl1_weight	0,993

Celková přesnost (anglicky accuracy) prediktivního modelu logistické regrese je 67,62 %. Senzitivita (anglicky true positive rate) pro hodnotu 0 je 67,15 %; specificita (anglicky true negative rate) pro výsledek 1 je 68,11 %. Pozitivní prediktivní hodnota (anglicky zkratka PPV) modelu pro hodnotu 0 je 69,01 %; negativní prediktivní hodnota (anglicky zkratka NPV) pro výsledek 1 je 66,22 % (viz tabulka č. 10).

Tab. 10: Confusion matice pro logistickou regresi

Výstup \ Cíl	Hráč 2	Hráč 1	Celkem
Hráč 2	**2169** 34,52 %	**974** 15,50 %	**3143** 69,01 % 30,99 %
Hráč 1	**1061** 16,88 %	**2080** 33,10 %	**3141** 66,22 % 33,78 %
Celkem	**3230** 67,15 % 32,85 %	**3054** 68,11 % 31,89 %	**4249 / 6284** 67,62 % 32,38 %

Poznámka: Confusion matice zaznamenává shody (zelená barva) a neshody (červená barva) výstupu modelu strojového učení (tj. predikce výsledku zápasu) a cíle (tj. reálného výsledku zápasu).

5.2.3 Sekvenční neuronová síť

Přesnost (anglicky accuracy) prediktivního modelu neuronové sítě je 67,33 %. Senzitivita (anglicky true positive rate) je 64,45 %; specificita (anglicky true negative rate) je 71,66 %. Pozitivní prediktivní hodnota (anglicky zkratka PPV) modelu je 77,35 %; negativní prediktivní hodnota (anglicky zkratka NPV) je 57,31 % (viz tabulka č. 11).

Tab. 11: Confusion matice pro sekvenční neuronovou síť

Výstup \ Cíl	Hráč 2	Hráč 1	Celkem
Hráč 2	**2431** 38,69 %	**712** 11,33 %	**3143** 77,35 % 22,65 %
Hráč 1	**1341** 21,34 %	**1800** 28,64 %	**3141** 57,31 % 42,69 %
Celkem	**3772** 64,45 % 35,55 %	**2512** 71,66 % 28,34 %	**4231 / 6284** 67,33 % 32,67 %

Poznámka: Confusion matice zaznamenává shody (zelená barva) a neshody (červená barva) výstupu modelu strojového učení (tj. predikce výsledku zápasu) a cíle (tj. reálného výsledku zápasu).

5.2.4 Úspěšnost vypsaných kurzů od bookmakerů

Přesnost kurzů vypsaných od bookmakerů je 66,79 %. Z celkových 6443 zápasů by tedy člověk při sázce na hráče s nižším kurzem vyhrál dohromady 4197 sázek a prohrál 2087 sázek (viz tabulka č. 12).

Tab. 12: Přesnost predikcí bookmakerů

Celkem zápasů	**6443**
Výhry	4197
Prohry	2087
Přesnost predikcí	**66,79 %**

5.2.5 Shrnutí výsledků prediktivních modelů

Nejvyšší celkovou přesnost má prediktivní model logistické regrese (67,62 %) a je tedy v našem případě (pro naše data) nejvhodnějším modelem pro retrospektivní analýzu ziskovosti sportovního sázení s konkrétními sázkovými strategiemi. Zároveň je přesnost tohoto modelu vyšší než přesnost vypsaných kurzů od bookmakerů (66,79 %).

Výchylka oproti ostatním modelům, která stojí za povšimnutí, je pozitivní prediktivní hodnota u naší neuronové sítě, která vychází na 77,35 %. Oproti tomu negativní prediktivní hodnota je pouhých 57,31 %. Vztah mezi hráči v obou případech (při outputu 0 i 1) by tak mohl být předmětem další analýzy a mohl by odhalit důležité

poznatky o tom, u kterých typů případů je daná neuronová síť nejpřesnější. U random forest a logistické regrese jsou pozitivní i negativní prediktivní hodnoty téměř stejné, pouze s mírnými odchylkami.

5.3 Výstupy jednotlivých sázkových strategií

5.3.1 Sázení s fixní částkou

5.3.1.1 Strategie A: Všechny zápasy podle kurzů

Při sázení podle kurzů od bookmakerů na všechny zápasy fixní částkou 500 byla celková výnosnost sázení -3,66 %. Na sázky byla dohromady vynaložena částka 3.142.000, přičemž výsledný čistý profit činí -115.004 (viz tabulka č. 13).

Tab. 13: Výstup strategie A

Součet sázek	3.142.000
Čistý zisk	-115.004
Výnosnost	-3,66 %
Sázek celkem	6.284
Sázek vyhráno	4.197
Sázek prohráno	2.087
Míra výher	66,79 %

5.3.1.2 Strategie B: Všechny zápasy podle predikcí

Při sázení podle predikcí modelu logistické regrese na všechny zápasy fixní částkou 500 byla celková výnosnost sázení 2,22 %. Na sázky byla dohromady vynaložena částka 3.142.000, přičemž výsledný čistý profit činí 69.620 (viz tabulka č. 14).

Tab. 14: Výstup strategie B

Součet sázek	3.142.000
Čistý zisk	69.620
Výnosnost	2,22 %
Sázek celkem	6.284
Sázek vyhráno	4.249
Sázek prohráno	2.035
Míra výher	67,62 %

5.3.1.3 Strategie C: Pravděpodobnostní sázení podle kurzů

Při sázení podle kurzů od bookmakerů s podmínkou sázky pouze na zápasy s kurzem 1,43 (odpovídá cca 70 % šanci na výhru) a méně fixní částkou 500 byla celková výnosnost sázení -3,3 %. Na sázky byla dohromady vynaložena částka 1.343.000, přičemž výsledný čistý profit činí -44.287 (viz tabulka č. 15). Ačkoliv tato strategie dokázala vyhrát 77,7 % všech sázek, zisková marže z kurzů 1,33 a méně je v průměru natolik nízká, že ani tak vysoká míra výher nedokáže přebít ztracené finance na prohrách.

Tab. 15: Výstup strategie C

Součet sázek	1.343.000
Čistý zisk	-44.287
Výnosnost	-3,3 %
Sázek celkem	2.686
Sázek vyhráno	2.087
Sázek prohráno	599
Míra výher	77,7 %

5.3.1.4 Strategie D: Pravděpodobnostní sázení podle predikcí

D1: Bez spodní hranice vypsaného kurzu

Při sázení podle predikcí modelu logistické regrese na zápasy s predikovanou pravděpodobností výhry 70 % hráče 1 nebo hráče 2 fixní částkou 500 byla celková výnosnost sázení 5,105 %. Na sázky byla dohromady vynaložena částka 1.312.000, přičemž výsledný čistý profit činí 66.980 (viz tabulka č. 16).

Tab. 16: Výstup strategie D1

Součet sázek	1.312.000
Čistý zisk	66.980
Výnosnost	5,1 %
Sázek celkem	2.624
Sázek vyhráno	2.114
Sázek prohráno	510
Míra výher	80,56 %

D2: Se spodní hranicí vypsaného kurzu

Při sázení podle predikcí modelu logistické regrese na zápasy s predikovanou pravděpodobností výhry 70 % hráče 1 nebo hráče 2 a omezením pouze na zápasy s kurzem 1,8 a více (u predikovaného výherce) fixní částkou 500 byla celková výnosnost sázení 65,63 %. Na sázky byla dohromady vynaložena částka 33.000, přičemž výsledný čistý profit činí 21.656 (viz tabulka č. 17).

Tab. 17: Výstup strategie D2

Součet sázek	33.000
Čistý zisk	21.656
Výnosnost	65,63 %
Sázek celkem	66
Sázek vyhráno	51
Sázek prohráno	15
Míra výher	77,27 %

5.3.2 Sázení s negativní progresí (Martingale systém)

5.3.2.1 Strategie E: Všechny zápasy podle kurzů

Při 100 pokusech promíchání zápasů a aplikování algoritmu s negativní progresí na všechny zápasy podle kurzů vypsaných od bookmakerů byla celková průměrná výnosnost sázení (při fixní částce sázky 500 na každý zápas) -8,06 %. Na sázky byla dohromady vynaložena částka 3.142.000, přičemž výsledný průměrný čistý profit činí -253.263 (viz tabulka č. 18).

Tab. 18: Výstup strategie E

Součet sázek	3.142.000
Čistý zisk	-253.263
Výnosnost	-8,06 %
Sázek celkem	6.284
Sázek vyhráno	4.197
Sázek prohráno	2.087
Míra výher	66,79 %

5.3.2.2 Strategie F: Všechny zápasy podle predikcí

Při 100 pokusech promíchání zápasů a aplikování algoritmu s negativní progresí na všechny zápasy podle predikcí modelu logistické regrese byla celková průměrná výnosnost sázení (při fixní částce sázky 500 na každý zápas) 4,39 %. Na sázky byla dohromady vynaložena částka 3.142.000, přičemž výsledný průměrný čistý profit činí 137.950 (viz tabulka č. 19).

Tab. 19: Výstup strategie F

Součet sázek	3.142.000
Čistý zisk	137.950
Výnosnost	4,39 %
Sázek celkem	6.284
Sázek vyhráno	4.249
Sázek prohráno	2.035
Míra výher	67,62 %

5.3.2.3 Strategie G: Pravděpodobnostní sázení podle kurzů

Při 100 pokusech promíchání zápasů a aplikování algoritmu s negativní progresí na zápasy podle kurzů vypsaných od bookmakerů (pouze zápasy, kde je mezi oběma kurzy maximální rozdíl 0,2) byla celková průměrná výnosnost sázení (při fixní částce sázky 500 na každý zápas) 20,69 %. Na sázky byla dohromady vynaložena částka 184.000, přičemž výsledný průměrný čistý profit činí 38.069 (viz tabulka č. 20).

Tab. 20: Výstup strategie G

Součet sázek	184.000
Čistý zisk	38.069
Výnosnost	20,69 %
Sázek celkem	368
Sázek vyhráno	193
Sázek prohráno	175
Míra výher	52,45 %

5.3.2.4 Strategie H: Pravděpodobnostní sázení podle predikcí

Při 100 pokusech promíchání zápasů a aplikování algoritmu s negativní progresí na zápasy (pouze ty, kde je predikovaná pravděpodobnost výhry hráče 1 a hráče 2 v rozmezí 48–52 %) podle predikcí modelu logistické regrese byla celková průměrná výnosnost sázení (při fixní částce sázky 500 na každý zápas) 41,13 %. Na sázky byla dohromady vynaložena částka 200.000, přičemž výsledný průměrný čistý profit činí 82.266 (viz tabulka č. 21).

Tab. 21: Výstup strategie H

Součet sázek	200.000
Čistý zisk	82.266
Výnosnost	41,13 %
Sázek celkem	400
Sázek vyhráno	191
Sázek prohráno	209
Míra výher	47,75 %

5.3.3 Shrnutí výsledků použitých algoritmů a sázkových strategií

Jak lze pozorovat v tabulce č. 22, v každé kategorii (A vs B, C vs D1 a D2, E vs F, G vs H) zvítězily strategie využívající predikce modelu logistické regrese. Strategie G a H se na první pohled mohou zdát úspěšné, nicméně výsledky těchto algoritmů jsou vysoce volatilní z důvodu využívání negativní progrese – jedná se o průměr 100 pokusů, přičemž mnoho z těchto pokusů v obou případech skončilo v záporných číslech. Výrazně úspěšnější, ale hlavně spolehlivější a stabilnější, je strategie D2, která nakonec svou celkovou výnosností sázek překonala všechny ostatní strategie.

Tab. 22: Srovnání výsledků zvolených sázkových strategií

VŠECHNY ZÁPASY S FIXNÍ ČÁSTKOU		
A	Sázení na všechny zápasy fixní částkou podle kurzů od ***bookmakerů***	-3,66 %
B	Sázení na všechny zápasy fixní částkou podle ***predikcí LR***	2,216 %
	ROZDÍL	5,88 %
VYBRANÉ ZÁPASY S FIXNÍ ČÁSTKOU		
C	Sázení na vybrané zápasy fixní částkou podle kurzů od ***bookmakerů***	-3,298 %
D1	Sázení na vybrané zápasy fixní částkou podle ***predikcí LR*** (bez spodní hranice vypsaného kurzu)	5,105 %
	ROZDÍL (C a D1)	8,4 %
D2	Sázení na vybrané zápasy fixní částkou podle ***predikcí LR*** (se spodní hranicí vypsaného kurzu)	65,625 %
	ROZDÍL (C a D2)	68,92 %
VŠECHNY ZÁPASY S NEGATIVNÍ PROGRESÍ		
E	Sázení na všechny zápasy s negativní progresí podle kurzů od ***bookmakerů***	-8,061 %
F	Sázení na všechny zápasy s negativní progresí podle ***predikcí LR***	4,391 %
	ROZDÍL	12,45 %
VYBRANÉ ZÁPASY S NEGATIVNÍ PROGRESÍ		
G	Sázení na vybrané zápasy s negativní progresí podle kurzů od ***bookmakerů***	20,69 %
H	Sázení na vybrané zápasy s negativní progresí podle ***predikcí LR***	41,133 %
	ROZDÍL	20,44 %

6 Diskuse

6.1 Diskuse k výsledkům práce

V rámci práce byl stanoven cíl vybrat nejpřesnější model strojového učení pro predikci výsledků tenisových zápasů. Pro tento cíl byly zvolen tři modely – rozhodovací strom (konkrétně „random forest“), logistická regrese a sekvenční neuronová síť. Všechny modely předčily prediktivní úspěšnost kurzů od bookmakerů, přičemž logistická regrese disponuje nejvyšší přesností pro náš konkrétní případ.

Z důvodu komplexity jednotlivých algoritmů však nelze jednoduše určit, z jakého důvodu byla logistická regrese nejúspěšnější. Jedním z logických důvodů však může být fakt, že predikce binárního výsledku sportovních utkání je ukázkový případ lineárního klasifikačního problému, což je přesně účel logistické regrese. Modely rozhodovacího stromu a neuronové sítě jsou naopak velmi obecné modely, které dokážou zpracovat téměř jakýkoliv úkol, takže se dá očekávat menší přesnost než u modelu specializovaného na konkrétní případ.

V další řadě bylo cílem zároveň retrospektivně srovnat výnosnost sázení na tenisové zápasy s vytvořenými predikce a s kurzy od bookmakerů. Byly vytvořeny celkem čtyři kategorie sázkových strategií – sázení na všechny zápasy fixní částkou, sázení na vybrané zápasy fixní částkou, sázení na všechny zápasy s negativní progresí a sázení na vybrané zápasy s negativní progresí. Každá kategorie pak obsahuje strategie s použitím kurzů od bookmakerů a strategie s použitím predikcí vytvořených modelem strojového učení.

Ve všech kategoriích jsou naše predikce vytvořené logistickou regresí výnosnější než kurzy (predikce) bookmakerů. A nejen že jsou výnosnější, ale výnosnost téměř všech strategií (až na strategie s vybranými zápasy a negativní progresí) je v případě bookmakerů v záporných číslech, zatímco v případě našich predikcí v kladných. Rozdíl ve výnosnosti v jednotlivých skupinách strategií se pohybuje v rozmezí cca 6–12 % a v případě strategií C a D2 dokonce cca 69 %, což jsou poněkud vysoká procenta, pakliže bereme v potaz to, že je přesnost predikcí našeho modelu pouze o 0,83 % vyšší než prediktivní přesnost kurzů od bookmakerů.

Nejlepších výsledků dosahuje strategie D2, jejíž charakter spočívá nejen v minimalizaci risku sázením na zápasy s vysokou pravděpodobností výhry, ale taky v podmínce sázení pouze na zápasy s určitou výší vypsaných kurzů pro maximalizaci zisku. Pro nejvýnosnější strategii je tedy zapotřebí propojit zároveň kurzy od bookmakerů a predikce vytvořené modelem strojového učení.

Je však také nutné dodat, že čím více (a čím přísnější) podmínek konkrétní zápasy v dané sázkové strategii musí splňovat, tím je zákonitě menší počet možných sázek. U strategie D2 bylo například vsazeno pouze na 66 zápásů z celkového počtu 6284 dostupných zápasů. Jelikož náš testovací vzorek zahrnoval zápasy od roku 2019 do

začátku roku 2022 (dohromady 38 měsíců), znamenalo by to, že podle této strategie můžeme v horizontu tří let sázet pouze zhruba 1,75 krát za měsíc. Pakliže by měl člověk zájem sázet více aktivně, bylo by nutné zvážit, zda se například při ohledu na výši dostupného kapitálu na sázení vyplatí podmínky sázkové strategie zmírnit, aby se počet dostupných sázek zvýšil, ale konsekvetně i potenciálně snížila procentuální výnosnost.

6.2 Srovnání výsledků s dalšími studiemi

Logistická regrese je hojně využívanou metodou strojového učení při predikování výsledků sportovních utkání a je využívána napříč všemi sporty.

Prasetio a Harlili (2016) použili logistickou regresi pro predikci výsledků fotbalových utkání, přičemž použili pouze čtyři statisticky nejvýznamnější nezávislé proměnné na základě výsledků z dalších studií. Dosáhli přesnosti 69,5 %.

Chenjie Cao (2012) použil pro predikci basketbalových utkání jak logistickou regresi, tak i ANN (umělé neuronové sítě). Obdobně jako v naší studii, logistická regrese dosáhla lepších výsledků než neuronová síť, a to konkrétně 69,67 % oproti 68,01 %. Autor zároveň uvádí, že je natrénovaný model praktický a dokáže s uvedenou přesností predikovat i výsledky ještě neodehraných zápasů.

Zaveri et al. (2018) ke své studii zvolili komplexnější přístup predikcí fotbalových zápasů. V první řadě použili, mimo jiné, stejně jako v naší studii, logistickou regresi, random forest a umělou neuronovou síť. Následně však predikce vytvořili na dvou datových setech. V prvním použili pouze data z již odehraných fotbalových zápasů. V druhém k historii zápasů přidali i historii jednotlivých týmů – pokud tedy tým A a tým B mají v historii proti sobě odehrané zápasy, statistiky z těchto zápasů jsou taktéž použity pro predikci výsledku jejich budoucích společných zápasů. Nejlepších výsledků však opět dosahuje logistická regrese, a to v obou případech. Při použití pouze dat z odehraných zápasů, neuronová síť dosáhla přesnosti 63,1 %, random forest 61,53 % a logistická regrese 63,94 %. Při použití jak dat z odehraných zápasů, tak historie jednotlivých týmů dosáhla neuronová síť 69,2 %, random forest 69,9 % a logistická regrese 71,63 %.

V review studii Horvata a Joba (2020) bylo srovnáno celkem 36 výzkumů na přesnost modelů strojového učení při predikci sportovních utkání. Výsledné přesnosti predikcí se v těchto výzkumech pohybují od 55,52 % do 93 %, a v průměru 72,96 %.

Z uvedených studií se tedy dá konstatovat hypotéza, že při predikci sportovních utkání je logistická regrese potenciálně nejpřesnějším modelem strojového učení. Nicméně je zaručeně vhodné vždy použít a otestovat více metod strojového učení, abychom s jistotou dosáhli co nejvyšší přesnosti predikcí pro naše konkrétní data.

6.3 Omezení práce a doporučení pro další výzkum

S automatizací predikcí sportovních utkání se pojí mnoho problémů, pro které zatím neexistuje žádné jednoduché řešení. Je například mnoho potenciálních proměnných, které zkušený sportovní fanoušek zvládne pečlivou analýzou vypozorovat ze sledování sportovních utkání, ale neexistuje žádný efektivní způsob, jak tyto proměnné kvantifikovat na historii například 30 tisíc zápasů. Může se jednat například o vliv zranění na hráče, kdy se mírně pozmění jeho styl hry. Takový případ může vytvořit slabiny, které soupeři s odlišným stylem hry mohou využít k výhře. Ve výsledku se tedy jedná o souboj mezi kvalitou a kvantitou. Modely strojového učení zvládnou vytvořit prediktivní model, který zanalyzuje desetitisíce zápasů během pár vteřin s relativně vysokou prediktivní přesností. Na druhou stranu zkušený sportovní fanoušek se zájmem o sportovní sázení zvládne zanalyzovat pouze pár zápasů denně, ale potenciálně s výrazně vyšší úspěšností. Největším problémem při zvyšování přesnosti prediktivních modelů strojového učení je tedy nalezení způsobu kvantifikace těchto zdánlivě nekvantifikovatelných proměnných.

Naší práci však jdou vytknout primárně dvě věci. V první řadě byla zjednodušena manipulace s daty při výpočtu historické míry výher všech hráčů. U každého zápasu je uvedena tato hodnota v konečné podobě – tedy pokud u posledního zápasu je míra výher hráče A 50 %, tak je takto uvedena i u jeho historicky prvního zápasu. Toto zjednodušení bylo nicméně zvoleno na základě hypotézy, že si hráči v průběhu svých kariér většinou udržují určitou míru výher (tzn. nemá tendenci se v průběhu kariéry drasticky měnit).

V další řadě jsme byli poučeni ze studie Zaveri et al. (2018), že by se přesnost našeho modelu dala potenciálně výrazně zvýšit při použití historie utkání jednotlivých hráčů proti sobě (tzv. „head to head“).

Co se širšího využití prediktivních modelů strojového učení a samotných predikcí výsledků sportovních utkání týče, v rámci sportovního managementu je možné nejen v oblasti sportovního sázení, ale i v oblasti lepší přípravy sportovců a týmů na budoucí utkání. Informační hodnota vypsaných kurzů na utkání může být potenciálně zkreslena marketingovými strategiemi konkrétních bookmakerů. Proto se na ně jako na ukazatele pravděpodobného výsledku zápasu nelze zaručeně spolehnout. V tom je schopnost predikovat výsledky zápasů na základě dat a faktů výrazně spolehlivější. Mimo samotný výsledek zápasu se pomocí modelů strojového učení dá předpovídat i průběh sportovních utkání. Sportovec tak může předem vědět, jaký bude pravděpodobně průběh a výsledek zápasu a adekvátně se na něj připravit skrze pečlivou analýzu silných stránek soupeře a následnou optimalizaci strategie tréninku.

7 Závěry

V naší studii byly použity dohromady tři modely strojového učení. Jako nejúspěšnější se ukázala být logistická regrese, která dosáhla přesnosti predikcí 67,62 %. Vedle toho byly modely random forest a sekvenční neuronová síť, které dosáhly téměř stejných výsledků, a to 67,04 % a 67,33 %.

Nejvyšší statistickou významnost měly u logistické regrese proměnné zastupující ATP rank hráče 2, průměrný vypsaný kurz na hráče 1 a 2, kariérní míru výher hráče 2 a kariérní míru výher na konkrétním povrchu kurtu hráče 1 a 2 (tedy pl2_rank, pl1_avg_bookmaker_odds, pl2_avg_bookmaker_odds, pl2_win_rate, pl1_swrate, pl2_swrate).

U modelu random forest mají nejvyšší významnost (měřeno atributem „importance“) nezávislé proměnné, stejně jako u logistické regrese, zastupující průměrný vypsaný kurz na hráče 1 a 2, kariérní míru výher na konkrétním povrchu kurtu hráče 1 a 2, ale navíc i rozdíl mezi kariérní mírou výher na konkrétním povrchu kurtu hráče 1 a 2 (tedy pl2_avg_bookmaker_odds, pl1_avg_bookmaker_odds, pl2_swrate, swrate_diff, pl1_swrate). U modelu sekvenční neuronové sítě nelze jednoznačně zjistit důležitost jednotlivých nezávislých proměnných.

Dá se tedy konstatovat, že obecně nejdůležitějšími prediktory jsou, mimo kurzů od bookmakerů, primárně údaje o kariérní míře výher daného hráče. Za další významný prediktor můžeme označit umístění hráčů na ATP žebříčku.

Při analýze vypsaných kurzů od bookmakerů bylo zjištěno, že sázení vždy na nižší kurz vyústí ve výhru v 66,79 % sázek na dostupné zápasy v testovacím vzorku. Přesnost našeho nejúspěšnějšího modelu je tedy pouze o 0,83 % vyšší než přesnost predikcí od bookmakerů.

I přes takto zdánlivě mizivý rozdíl v přesnosti predikcí je však výdělečnost našeho modelu relativně výrazně vyšší než výdělečnost při sázení podle kurzů od bookmakerů. Ve všech použitých sázkových strategiích vždy zvítězila varianta používající predikce vytvořeného modelu logistické regrese – napříč kategoriemi byly tyto varianty výnosnější o 5,88 %, 8,4 %, 12,45 % a 20,44 % než jejich protějšky využívající pouze vypsaných kurzů od bookmakerů.

Nejúspěšnější byla však strategie D2, která byla oproti strategii C výnosnější o 65,625 % – tato strategie totiž nevyužívala pouze predikcí vytvořených modelem logistické regrese, ale zároveň byly do podmínky sázení započteny i kurzy od bookmakerů, aby se sázelo pouze na ty sázky, kde je jak vysoká pravděpodobnost úspěchu, tak i vysoká míra návratnosti vsazené částky.

Na závěr práce tedy lze konstatovat, že nejúspěšnějším modelem strojového učení pro naše konkrétní data tenisových zápasů je logistická regrese a nejvýnosnější sázková strategie je taková, která do podmínek započítá nejen predikce vytvořené modelem strojového učení, ale zároveň i kurzy vypsané od bookmakerů.

Použité zdroje

Amazon Web Services. (n.d.). *What is Overfitting? - Overfitting in Machine Learning Explained - AWS*. Retrieved 9 April 2024, from https://aws.amazon.com/what-is/overfitting/

Ardeljan, E. (2023). Impact of Sports Gambling Legality on U.S. States' Real GDP per Capita. *Williams Honors College, Honors Research Projects*. https://ideaexchange.uakron.edu/honors_research_projects/1729

Baştanlar, Y., & Özuysal, M. (2013). Introduction to Machine Learning. In M. Yousef & J. Allmer (Eds.), *miRNomics: MicroRNA Biology and Computational Analysis* (pp. 105–128). Humana Press. https://doi.org/10.1007/978-1-62703-748-8_7

Bhandari, P. (2022, February 3). *Independent vs. Dependent Variables | Definition & Examples*. Scribbr. https://www.scribbr.com/methodology/independent-and-dependent-variables/

Birch, K., Cochrane, D., & Ward, C. (2021). Data as asset? The measurement, governance, and valuation of digital personal data by Big Tech. *Big Data & Society*, *8*(1), 205395172110173. https://doi.org/10.1177/20539517211017308

Cantagallo, E. (n.d.). *ATP Tennis Data with betting odds*. Kaggle. Retrieved 4 April 2024, from https://www.kaggle.com/datasets/edoardoba/atp-tennis-data?resource=download

Cao, C. (2012). Sports Data Mining Technology Used in Basketball Outcome Prediction. *Dissertations*. https://arrow.tudublin.ie/scschcomdis/39

Castrounis, A. (2017). *What Is Data Science, and What Does a Data Scientist Do?* KDnuggets. https://www.kdnuggets.com/what-is-data-science-and-what-does-a-data-scientist-do

Charpentier, A. (2019, April 18). A brief history of sports betting [Billet]. *Freakonometrics*. https://doi.org/10.58079/ovd7

Chóliz, M. (2018). Ethical gambling: A necessary new point of view of gambling in public health policies. *Frontiers in Public Health*, *6*, 12. https://doi.org/10.3389/fpubh.2018.00012

Copeland, B.J. (n.d.). *Artificial intelligence—Reasoning, Algorithms, Automation | Britannica*. Retrieved 9 April 2024, from https://www.britannica.com/technology/artificial-intelligence/Reasoning

Dietrich, D., Heller, B., Yang, B., & EMC Education Services (Eds.). (2015). *Data science & big data analytics: Discovering, analyzing, visualizing and presenting data*. Wiley.

Hintze, A. (2016, November 14). *Understanding the four types of AI, from reactive robots to self-aware beings*. The Conversation. http://theconversation.com/understanding-the-four-types-of-ai-from-reactive-robots-to-self-aware-beings-67616

Horvat, T., & Job, J. (2020). The use of machine learning in sport outcome prediction: A review. *WIREs Data Mining and Knowledge Discovery*, *10*(5), e1380. https://doi.org/10.1002/widm.1380

IBM. (n.d.). *What Is Random Forest? | IBM*. Retrieved 9 April 2024, from https://www.ibm.com/topics/random-forest

Johnson, D. (2024, March 16). *What is Data Science? Introduction, Basic Concepts & Process*. https://www.guru99.com/data-science-tutorial.html

Joshi, N. (2019, June 19). *7 Types Of Artificial Intelligence*. Forbes. https://www.forbes.com/sites/cognitiveworld/2019/06/19/7-types-of-artificial-intelligence/

Kitchin, R., & McArdle, G. (2016). What makes big data, big data? Exploring the ontological characteristics of 26 datasets. *Big Data & Society*, *3*(1), 205395171663113. https://doi.org/10.1177/2053951716631130

Klein, C. (2021, July 30). *5 Myths About the Ancient Olympics*. History. https://www.history.com/news/5-myths-about-the-ancient-olympics

Klein, J., & Giglioni, G. (2020). Francis Bacon. In E. N. Zalta (Ed.), *The Stanford Encyclopedia of Philosophy* (Fall 2020). Metaphysics Research Lab, Stanford University. https://plato.stanford.edu/archives/fall2020/entries/francis-bacon/

Knowledge at Wharton. (1999). *Mining Data for Nuggets of Knowledge*. https://knowledge.wharton.upenn.edu/article/mining-data-for-nuggets-of-knowledge/

Lawton, G. (2023, August 30). *What Is Regression in Machine Learning?* TechTarget. https://www.techtarget.com/searchenterpriseai/feature/What-is-regression-in-machine-learning

Matheson, V. (2021). An overview of the economics of sports gambling and an introduction to the symposium. *Eastern Economic Journal*, *47*(1), 1–8. https://doi.org/10.1057/s41302-020-00182-4

Oddspedia. (2023). *Fibonacci Betting System—Reliable Sports Betting Strategy?* https://oddspedia.com/betting/strategies-systems/fibonacci-system-explained

Pempus, B. (2024, February 26). *States Where Sports Betting Is Legal*. Forbes Betting. https://www.forbes.com/betting/legal/states-where-sports-betting-is-legal/

Prasetio, D., & Harlili, Dra. (2016). Predicting football match results with logistic regression. *2016 International Conference On Advanced Informatics: Concepts, Theory And Application (ICAICTA)*, 1–5. https://doi.org/10.1109/ICAICTA.2016.7803111

Press, G. (2013). *A Very Short History Of Data Science*. Forbes. https://www.forbes.com/sites/gilpress/2013/05/28/a-very-short-history-of-data-science/

Pykes, K. (2022, November). *Sports Analytics: How Different Sports Use Data Analytics*. https://www.datacamp.com/blog/sports-analytics-how-different-sports-use-data-analysis

Saha, D. (2020). *Google Cloud BrandVoice: How The World Became Data-Driven, And What's Next*. Forbes. https://www.forbes.com/sites/googlecloud/2020/05/20/how-the-world-became-data-driven-and-whats-next/

Sarker, I. H. (2021a). Data science and analytics: An overview from data-driven smart computing, decision-making and applications perspective. *SN Computer Science*, *2*(5), 377. https://doi.org/10.1007/s42979-021-00765-8

Sarker, I. H. (2021b). Deep learning: A comprehensive overview on techniques, taxonomy, applications and research directions. *SN Computer Science*, *2*(6), 420. https://doi.org/10.1007/s42979-021-00815-1

Sarker, I. H. (2021c). Machine learning: Algorithms, real-world applications and research directions. *SN Computer Science*, *2*(3), 160. https://doi.org/10.1007/s42979-021-00592-x

Sohail, S. (2024). *Sports Betting Odds: How They Work and How to Read*. Investopedia. https://www.investopedia.com/articles/investing/042115/betting-basics-fractional-decimal-american-moneyline-odds.asp

Srakocic, S. (n.d.). *What Jobs are Available in Sports Management?* All Business Schools. Retrieved 10 April 2024, from https://www.all-businessschools.com/sports-management/job-description/

Statista. (2020, April). *Gross Gaming Revenue as a share of GDP by European country*. https://www.statista.com/statistics/967875/gross-gambling-revenue-share-gdp-europe-by-country/

Tuček, J., & Dolinová, M. (2001). *Kasina aneb Co nevíte o nejrychlejším hazardu*. https://www.databazeknih.cz/knihy/kasina-aneb-co-nevite-o-nejrychlejsim-hazardu-124045

Tukey, J. W. (1962). The Future of Data Analysis. *The Annals of Mathematical Statistics*, *33*(1), 1–67. JSTOR.

Zaki, M. J., & Meira, W. (2014). *Data mining and analysis: Fundamental concepts and algorithms*. Cambridge university press.

Zarra, R. (2023, March 5). *Machine Learning: Linearity vs Nonlinearity | LinkedIn*. LinkedIn. https://www.linkedin.com/pulse/machine-learning-linearity-vs-nonlinearity-reday-zarra/

Zaveri, N., Shah, U., Tiwari, S., Shinde, P., & Teli, L. K. (2018). Prediction of football match score and decision making process. *International Journal on Recent and Innovation Trends in Computing and Communication*, *6*(2).

Příloha A Postup manipulace s daty

A.1 Transformace a čištění originálních dat

Jako první krok byla v jazyce Python naimportována knihovna *pandas*, ze které byly čerpány funkce pro manipulaci s daty.

```
# import knihovny pandas
import pandas as pd
```

Datový soubor s tenisovými zápasy je ve formátu CSV, takže pro jeho otevření byla využita funkce *read_csv* z knihovny *pandas* a soubor uložen do proměnné *df* (zkratka pro „dataframe“).

```
# uložení souboru tennis_data.csv do proměnné df
df = pd.read_csv('tennis_data.csv')
```

Pro základní přehled o sloupcích a hodnotách v nich se nacházejících byl využit programovací jazyk R, a to konkrétně funkce str(), jejímž výstupem jsou dohromady tři sloupce – první reprezentuje název jednotlivých sloupců v tabulce, druhý pak datový typ hodnot v těchto sloupcích a třetí obsahuje vždy prvních několik hodnot v každém sloupci. Obdobné funkce existují i v knihovně *pandas*, ale žádná z nich neumí vypsat datový typ a příklad hodnot zároveň. Níže je zobrazen syntax v jazyce R i výstup této funkce.

```
# uložení souboru tennis_data.csv to proměnné data a zavolání funkce str
data <- read.csv("tennis_data.csv")
str(data)

# výstup funkce str()
'data.frame':36120 obs. of  54 variables:
 $ ATP         : int  1 1 1 1 1 1 1 1 1 1 ...
 $ Location    : chr  "Adelaide" "Adelaide" "Adelaide" "Adelaide" ...
 $ Tournament  : chr  "Adelaide International 1" "Adelaide International 1"
"Adelaide International 1" "Adelaide International 1" ...
 $ Date        : chr  "2022-01-03" "2022-01-03" "2022-01-03" "2022-01-03" ...
 $ Series      : chr  "ATP250" "ATP250" "ATP250" "ATP250" ...
 $ Court       : chr  "Outdoor" "Outdoor" "Outdoor" "Outdoor" ...
 $ Surface     : chr  "Hard" "Hard" "Hard" "Hard" ...
 $ Round       : chr  "1st Round" "1st Round" "1st Round" "1st Round" ...
 $ Best.of     : int  3 3 3 3 3 3 3 3 3 3 ...
 $ Winner      : chr  "Kwon S.W." "Monteiro T." "Djere L." "Johnson S." ...
 $ Loser       : chr  "Nishioka Y." "Altmaier D." "Carballes Baena R." "Vukic A."
...
 $ WRank       : num  53 89 52 85 92 125 113 94 90 43 ...
 $ LRank       : num  81 84 79 156 103 59 40 64 137 83 ...
 $ WPts        : num  1115 805 1131 812 797 ...
 $ LPts        : num  823 813 837 440 740 ...
 $ W1          : num  6 6 7 6 7 6 6 7 6 6 ...
 $ L1          : num  1 2 5 4 6 4 3 5 2 4 ...
 $ W2          : num  6 3 7 2 6 6 6 7 6 4 ...
```

```
 $ L2          : num  2 6 6 6 4 7 4 5 4 6 ...
 $ W3          : num  NA 7 NA 6 NA 6 NA NA NA 6 ...
 $ L3          : num  NA 6 NA 4 NA 3 NA NA NA 4 ...
 $ W4          : num  NA NA NA NA NA NA NA NA NA NA ...
 $ L4          : num  NA NA NA NA NA NA NA NA NA NA ...
 $ W5          : num  NA NA NA NA NA NA NA NA NA NA ...
 $ L5          : num  NA NA NA NA NA NA NA NA NA NA ...
 $ Wsets       : num  2 2 2 2 2 2 2 2 2 2 ...
 $ Lsets       : num  0 1 0 1 0 1 0 0 0 1 ...
 $ Comment     : chr  "Completed" "Completed" "Completed" "Completed" ...
 $ B365W       : num  1.61 2.3 1.66 2 2.75 2 2.37 2 2.75 1.33 ...
 $ B365L       : num  2.3 1.61 2.2 1.8 1.44 1.8 1.57 1.8 1.44 3.4 ...
 $ PSW         : num  1.7 2.49 1.81 2.21 2.66 2.19 2.91 2.15 2.87 1.35 ...
 $ PSL         : num  2.26 1.6 2.1 1.74 1.53 1.75 1.46 1.78 1.47 3.45 ...
 $ MaxW        : num  1.76 2.6 1.83 2.3 2.81 2.25 2.91 2.15 3.32 1.35 ...
 $ MaxL        : num  2.5 1.62 2.26 1.8 1.54 1.8 1.57 1.82 1.47 3.6 ...
 $ AvgW        : num  1.64 2.38 1.7 2.1 2.64 2.09 2.62 2.04 2.89 1.32 ...
 $ AvgL        : num  2.22 1.57 2.12 1.72 1.47 1.73 1.48 1.77 1.41 3.35 ...
 $ EXW         : chr  "" "" "" "" ...
 $ EXL         : num  NA NA NA NA NA NA NA NA NA NA ...
 $ LBW         : num  NA NA NA NA NA NA NA NA NA NA ...
 $ LBL         : num  NA NA NA NA NA NA NA NA NA NA ...
 $ SJW         : num  NA NA NA NA NA NA NA NA NA NA ...
 $ SJL         : num  NA NA NA NA NA NA NA NA NA NA ...
 $ UBW         : num  NA NA NA NA NA NA NA NA NA NA ...
 $ UBL         : num  NA NA NA NA NA NA NA NA NA NA ...
 $ pl1_flag    : chr  "KOR" "BRA" "SRB" "USA" ...
 $ pl1_year_pro: num  2015 2011 2013 2012 2016 ...
 $ pl1_weight  : num  72 78 80 86 71 76 84 75 71 78 ...
 $ pl1_height  : num  180 183 185 188 175 191 183 183 183 185 ...
 $ pl1_hand    : chr  "Right-Handed" "Left-Handed" "Right-Handed" "Right-Handed"
...
 $ pl2_flag    : chr  "JPN" "GER" "ESP" "AUS" ...
 $ pl2_year_pro: num  2014 2014 2011 2018 2020 ...
 $ pl2_weight  : num  64 80 76 85 77 78 82 82 84 92 ...
 $ pl2_height  : num  170 188 180 188 188 185 188 183 183 198 ...
 $ pl2_hand    : chr  "Left-Handed" "Right-Handed" "Right-Handed" "Right-Handed"
...
```

Následně bylo potřeba rozhodnout, které prediktory chceme v naší analýze použít. V další řadě pak musíme zjistit, zda jsou námi zvolené prediktory ve správném formátu.

Z originálních dat využijeme prediktory *Series*, *Court*, *Surface*, *Round*, *Best of*, *Wrank*, *Lrank*, *Wpts*, *Lpts*, *pl1_year_pro*, *pl1_weight*, *pl1_height*, *pl1_hand*, *pl2_year_pro*, *pl2_weight*, *pl2_height*, *pl2_hand*.

Sloupce *B635W* až *UBL* obsahují kurzy vypsané bookmakery na konkrétní zápas. Tato informace je pro nás klíčová nejen jakožto prediktor pro strojové učení, ale zároveň i pro analýzu ziskovosti sázkových strategií využívající právě ony vypsané kurzy. Jelikož však ne každý zápas obsahuje kurz od všech bookmakerů dostupných v této tabulce (viz hodnoty *NA*), byly vytvořeny dva sloupce, které průměrují všechny dostupné kurzy. Tyto sloupce byly pojmenovány *pl1_avg_bookmaker_odds* a *pl2_avg_bookmaker_odds*.

```
# využití funkce mean() pro výpočet nových sloupců
```

```
df['pl1_avg_bookmaker_odds'] = df[['B365W', 'PSW', 'LBW', 'SJW',
'UBW']].mean(axis=1)

df['pl2_avg_bookmaker_odds'] = df[['B365L', 'PSL', 'LBL', 'SJL',
'UBL']].mean(axis=1)
```

Mimo námi zvolené prediktory byly zachovány sloupce *Date*, *Winner* a *Loser* – pro samotnou predikci jich sice nepoužijeme, ale budou nám sloužit pro přehlednost a následnou manipulaci s daty.

Ostatní sloupce jsou pro nás buď irelevantní, anebo se týkají údajů, které byly zaznamenány až během zápasu. Například sloupce *W1* až *L5*, které zaznamenávají počet vyhraných gamů v každém setu. Dále taky *Wsets* a *Lsets*, které udávají celkový počet vyhraných setů. Jelikož je cílem našich modelů strojového učení predikovat výsledky ještě neodehraných zápasů, nemá smysl ponechat údaje, které se zaznamenají až během jejich průběhu.

```
# odstranění irelevantních sloupců pomocí funkce drop()
df = df.drop(['ATP', 'Location', 'Tournament', 'W1', 'L1', 'W2', 'L2', 'W3',
'L3', 'W4', 'L4', 'W5', 'L5', 'Wsets', 'Lsets', 'B365W', 'B365L', 'PSW', 'PSL',
'MaxW', 'MaxL', 'AvgW', 'AvgL', 'EXW', 'EXL', 'LBW', 'LBL', 'SJW', 'SJL', 'UBW',
'UBL', 'pl1_flag', 'pl2_flag'], axis=1)
```

Dále bylo nutné data vyčistit. Záměrně byl doposud zanechán sloupec *Comment*, ve kterém je uvedeno, zda byl zápas řádně dokončen, nebo byl z jakéhokoliv důvodu (zranění apod.) předčasně ukončen. Pro naši predikci nás zajímají pouze zápasy s hodnotou „Completed“.

```
# filtr pouze těch řádků, které obsahují hodnotu ‚Completed‘
df = df[df['Comment'] == 'Completed']

# odstranění již nepotřebného sloupce Comment
df = df.drop(['Comment'], axis=1)
```

Dále bylo zapotřebí provést kódování u sloupců, které byly formátu „chr“ (text). Jedná se o sloupce *Series*, *Court*, *Surface*, *Round*, *pl1_hand* a *pl2_hand*. Převedením textové hodnoty na numerický kód byla data připravena na práci s prediktivními modely. Tento proces byl proveden s pomocí funkce *astype()*, a to následujícím kódem:

```
# pomocná proměnná s dvojicemi nového a původního názvu sloupců
col =
[['series_code','Series'],['court_code','Court'],['round_code','Round'],['surface
_code','Surface'],['best_of','Best
of'],['pl1_hand_code','pl1_hand'],['pl2_hand_code','pl2_hand']]

# kódování jednotlivých sloupců
for i in range(len(col)):
    df[col[i][0]] = df[col[i][1]].astype('category').cat.codes
```

Jakmile byl pro každý textový sloupec vytvořen odpovídající numerický, pro textové sloupce již není užití.

```
# odstranění textových sloupců
df = df.drop(['Series', 'Court', 'Surface', 'Round', 'Best of', 'pl1_hand',
'pl2_hand'], axis=1)
```

Poslední problematický sloupec, který zbyl, byl sloupec *Date*. Jak si můžeme všimnou ve výstupu funkce *str()* výše, byl tento sloupec datového typu *chr*. Jelikož budeme chtít rozdělit trénovací a testovací vzorek dat podle data konání zápasů, bylo potřeba je převést na datový typ *datetime* do nového sloupce s názvem *date*.

```
# změna formátu z chr na datetime do nového sloupce date
df['date'] = pd.to_datetime(df['Date'], format='%d/%m/%Y')

# odstranění původního sloupce Date
df = df.drop(['Date'], axis=1)
```

Přeposledním krokem čištění dat byly sjednoceny názvy všech sloupců – doposud se o jednotlivých hráčích referovalo jako o *Winner*/*Loser*, či *Player1*/*Player2*. Pro účel přehlednosti dat se budeme držet jmen *Player1* a *Player2* (nebo *pl1* a *pl2*):

```
# přejmenování sloupců pomocí funkce rename()
df.rename(columns={'Winner':'pl1', 'Loser':'pl2', 'WRank':'pl1_rank',
'LRank':'pl2_rank', 'WPts':'pl1_pts', 'LPts':'pl2_pts'}, inplace=True)
```

Posledním krokem bylo odstranění všech řádků, ve kterých se nacházejí odlehlé nebo prázdné hodnoty (*NA)*:

```
# odstranění odlehlých hodnot ve výšce a váze obou hráčů
df = df.drop(df[df['pl1_weight'] < 50].index)
df = df.drop(df[df['pl1_height'] < 100].index)
df = df.drop(df[df['pl1_height'] > 220].index)
df = df.drop(df[df['pl2_weight'] < 50].index)
df = df.drop(df[df['pl2_height'] < 100].index)
df = df.drop(df[df['pl2_height'] > 220].index)

# odstranění odlehlých hodnot ve vypsaných kurzech bookmakerů
df = df.drop(df[df['pl1_avg_bookmaker_odds'] > 20].index)
df = df.drop(df[df['pl2_avg_bookmaker_odds'] > 20].index)
df = df.drop(df[(df['pl1_avg_bookmaker_odds'] > 5) &
(df['pl2_avg_bookmaker_odds'] > 5)].index)

# odstranění řádků s prázdnými hodnotami
df = df.dropna()
```

A.2 Tvorba nových prediktorů

Logickým prediktorem, pro který lze předpovídat vysokou statistickou významnost, je kariérní míra výher každého hráče.

První řadě bylo potřeba si vytvořit dva nové datové sety (*wins* a *losses*) – první, který seskupí jednotlivé výherce a spočítá počet jejich výher a druhý, který seskupí

hráče, kteří prohráli, a spočítá počet jejich proher. Toho bylo docíleno funkcí *value_counts()*:

```
wins = df['pl1'].value_counts().reset_index()
losses = df['pl2'].value_counts().reset_index()
```

Pro následnou přehlednější manipulaci byly u obou datasetů pojmenovány první sloupec se jménem hráče jako *Player* a druhý *Wins*, respektive *Losses*:

```
wins.columns = ['Player', 'Wins']
losses.columns = ['Player', 'Losses']
```

Následně byly tyto nové datasety spojeny s pomocí funkce *merge()* s tím, že jsme datasety spojili na sloupci *Player* s operací „outer join" pro zahrnutí všech hráčů z obou datasetů:

```
player_stats = pd.merge(wins, losses, on='Player', how='outer').fillna(0)
```

V dalším kroku byl přidán sloupec s výpočtem míry výher vůči prohrám vydělením celkového počtu výher celkovým počtem her:

```
player_stats['Win_Rate'] = player_stats['Wins'] / (player_stats['Wins'] +
player_stats['Losses'])
```

Potenciální hodnotnou informací je i celkový počet odehraných zápasů v kariéře. Například hráči s žadnými výhrami a pouze jednou prohrou mají 0 % míru výher, ale na základě toho, že mají pouze jeden odehraný zápas, se jedná o zkreslenou informaci – je tedy vhodné algoritmu poskytnou i informaci právě o celkovém počtu odehraných zápasů. Pro tento účel byl vytvořen parametr *total_games* sečtením celkového počtu výher a proher:

```
player_stats['total_games'] = player_stats['Wins'] + player_stats['Losses']
```

Míru výher a počet odehraných her v kariéře z datasetu *player_stats* nyní můžeme spojit s naším původním datasetem *df*. Byly tak vytvořeny dohromady čtyři nové prediktory (krok 1 a 2) – jeden pro míru výher hráče 1 (*Winner*), druhý pro míru výher hráče 2 (*Loser*), třetí pro celkový počet odehraných zápasů hráče 1 (*Winner*) a čtvrtý pro celkový počet odehraných zápasů hráče 2 (*Loser*). Nasledně byly odstraněny přebytečné sloupce vytvořené procesem *merge*, a to *Player_x* a *Player_y* (krok 3) a nové prediktory přejmenovány, aby odpovídaly naší zvolené konvenci jmen (krok 4):

1. `df = pd.merge(df, player_stats[['Player', 'Total_games', 'Win_Rate']], left_on='pl1', right_on='Player', how='left')`

2. `df = pd.merge(df, player_stats[['Player', 'Total_games', 'Win_Rate']], left_on='pl2', right_on='Player', how='left')`

3. `df = df.drop(['Player_x', 'Player_y'], axis=1)`

```
4. df.rename(columns={'Win_Rate_x':'pl1_win_rate',
   'Win_Rate_y':'pl2_win_rate', 'total_games_x':'pl1_total_games',
   'total_games_y':'pl2_total_games'}, inplace=True)
```

Provedli jsme tedy spojení dvou datasetů – *df* a *player_stats* (u kterého jsme vybrali pouze sloupce *Player*, *Total_games* a *Win_rate*) – a to na sloupcích *Winner*/*Loser* z levého datasetu a sloupci *Player* z pravého. Zároveň jsme použili „left join", abychom zachovali pouze hráče z našeho originálního datasetu *df*.

V další řadě mimo míru výher obou soupeřů v každém zápase může být relevantní informací míra výher na konkrétním povrchu – mnoho hráčů je či v historii bylo proslulých svojí povrchovou dominancí. Postup bude podobný. Nejdříve byly vytvořeny dva datasety, u kterých byla použita funkce *groupby()* na *pl1* a *surface_code* (krok 1 a 2). Ty byly následně spojeny na parametrech *pl1*/*pl2* a *surface_code* (krok 3) za účelem výpočtu míry výher na konkrétním povrchu (krok 4):

```
1. surface_win_counts = df.groupby(['pl1',
   'surface_code']).size().reset_index(name='wins')

2. surface_lose_counts = df.groupby(['pl2',
   'surface_code']).size().reset_index(name='losses')

3. merged_counts = pd.merge(surface_win_counts, surface_lose_counts,
   left_on=['pl1', 'surface_code'], right_on=['pl2', 'surface_code'],
   how='outer')

4. merged_counts['surface_win_rate'] = merged_counts['wins'] /
   (merged_counts['losses'] + merged_counts['wins'])
```

V poslední řadě byly přidány sloupce pro povrchovou míru výher pro oba *pl1* i *pl2* v našem originálním datasetu *df* (krok 1 a 2) a přejmenovány, aby odpovídaly naší zavedené konvenci jmen (krok 3).

```
1. df = pd.merge(df, merged_counts[['pl1', 'surface_code',
   'surface_win_rate']], left_on=['pl1', 'surface_code'], right_on=['pl1',
   'surface_code'], how='left')

2. df = pd.merge(df, merged_counts[['pl2', 'surface_code',
   'surface_win_rate']], left_on=['pl2', 'surface_code'], right_on=['pl2',
   'surface_code'], how='left')

3. df.rename(columns={'surface_win_rate_x':'pl1_swrate',
   'surface_win_rate':'pl2_swrate'}, inplace=True)
```

Poslední čtyři prediktory, které mohou být pro modely strojového učení relevantní, jsou rozdíly mezi jednotlivými hráči, a to konkrétně rozdíl mezi parametry *rank*, *pts* (points), *win_rate* a *surface_win_rate*. Pro tento výpočet byla použita funkce z knihovny Pandas *apply()* a výchozí funkce jazyka Python *lambda*:

```
df['rank_diff'] = df.apply(lambda row: max(row['pl1_rank'], row['pl2_rank']) -
min(row['pl1_rank'], row['pl2_rank']), axis=1)
```

```
df['pts_diff'] = df.apply(lambda row: max(row['pl1_pts'], row['pl2_pts']) -
min(row['pl1_pts'], row['pl2_pts']), axis=1)

df['wrate_diff'] = df.apply(lambda row: max(row['pl1_win_rate'],
row['pl2_win_rate']) - min(row['pl1_win_rate'], row['pl2_win_rate']), axis=1)

df['swrate_diff'] = df.apply(lambda row: max(row['pl1_swrate'],
row['pl2_swrate']) - min(row['pl1_swrate'], row['pl2_swrate']), axis=1)
```

Po výstupu funkce *tolist()* (která zobrazí jména všech sloupců v datasetu) vidíme, že nyní disponujeme sloupci *pl1_total_games*, *pl1_win_rate*, *pl2_total_games*, *pl2_win_rate*, *pl1_swrate*, *pl2_swrate*, *rank_diff*, *pts_diff*, *wrate_diff*, *swrate_diff*. Dohromady tedy bylo vytvořeno deset nových prediktorů.

```
print(df.columns.tolist())

#výstup funkce tolist()
['pl1', 'pl2', 'pl1_rank', 'pl2_rank', 'pl1_pts', 'pl2_pts', 'pl1_year_pro',
'pl1_weight', 'pl1_height', 'pl2_year_pro', 'pl2_weight', 'pl2_height',
'pl1_avg_bookmaker_odds', 'pl2_avg_bookmaker_odds', 'series_code', 'court_code',
'surface_code', 'round_code', 'pl1_hand_code', 'pl2_hand_code', 'date',
'best_of', 'pl1_total_games', 'pl1_win_rate', 'pl2_total_games', 'pl2_win_rate',
'pl1_swrate', 'pl2_swrate', 'rank_diff', 'pts_diff', 'wrate_diff', 'swrate_diff']
```

V procesu přidávání nových prediktorů mohlo dojít k výskytu nových prázdných hodnot (*NA*) – byly tedy preventivně odstraněny:

```
df = df.dropna()
```

A.3 Tvorba a vyvážení závislé proměnné

Mimo prediktory bylo potřeba vytvořit závislou proměnnou, která bude později objektem predikcí. V našem případě chceme předpovídat výsledky tenisových utkání, a proto právě tato informace bude závislou proměnnou. Dataset *df* je doposud postaven tak, že *pl1* je vždy výherce. Byl tedy vytvořen nový sloupec *result*, kterému byla přiřazena hodnota *1* (pro *player1*; v případě výhry *player2* by hodnota byla *0*):

```
df['result'] = 1
```

Aby bylo trénování modelů strojového učení možné a optimální, bylo potřeba zastoupit oba potenciální binární výsledky zápasů ve stejné míře. Toho bylo docíleno rozdělením datasetu *df* na dvě poloviny podle *id* řádků s lichými a sudými hodnotami (krok 2), následného přehození všech hodnot pro *pl1* a *pl2* v „liché“ polovině datasetu *df* (krok 4), změny hodnoty *result* z *1* na *0* (krok 5) a finálního spojení obou polovin zpět dohromady (krok 6).

```
1. #tvorba sloupce s hodnotou ‚id‘
   df['id'] = range(1, len(df) + 1)

2. even_df = df[df['id'] % 2 == 0]
   odd_df = df[df['id'] % 2 != 0]

3. #pomocná proměnná obsahující dvojice hodnot k prohození
   values_to_swap =
   [['pl1','pl2'],["pl1_hand_code","pl2_hand_code"],["pl1_rank","pl2_rank"],[
   "pl1_pts","pl2_pts"],["pl1_year_pro","pl2_year_pro"],["pl1_weight","pl2_we
   ight"],["pl1_height","pl2_height"],['pl1_avg_bookmaker_odds','pl2_avg_book
   maker_odds'],['pl1_total_games','pl2_total_games'],['pl1_win_rate','pl2_wi
   n_rate'],['pl1_swrate','pl2_swrate']]

4. for i in range(len(values_to_swap)):
         odd_df[values_to_swap[i][0]], odd_df[values_to_swap[i][1]] =
         odd_df[values_to_swap[i][1]], odd_df[values_to_swap[i][0]]

5. odd_df['result'] = 0

6. final_df = pd.concat([even_df, odd_df], ignore_index=True)

   #seřazení výsledného datasetu podle parametru ‚id‘
   final_df = final_df.sort_values(by='id')
```

Příloha B Postup trénování prediktivních modelů

B.1 Příprava proměnných a vzorků dat

Import knihovny *Pandas* pro manipulaci s daty a *preprocessing* z knihovny *Sklearn* pro standardizaci dat:

```
import pandas as pd
from sklearn import preprocessing
```

Nastavení proměnných a vzorků dat a standardizace dat:

```
# nastavení pracovního datasetu
df = pd.read_csv("output.csv")

# nastavení prediktorů
predictors =
["surface_code","pl1_hand_code","pl2_hand_code","round_code","court_code","series
_code","best_of","pl1_rank","pl2_rank","pl1_pts","pl2_pts","pl1_year_pro","pl2_ye
ar_pro","pl1_weight","pl2_weight","pl1_height","pl2_height','pl1_avg_bookmaker_od
ds','pl2_avg_bookmaker_odds','pl1_total_games','pl1_win_rate','pl2_total_games','
pl2_win_rate','pl1_swrate','pl2_swrate','rank_diff','pts_diff','wrate_diff','swra
te_diff']

# tvorba train a test vzorků dat
train = df[df["date"] < "2019-01-01"]
test = df[df["date"] >= "2019-01-01"]

# rozlišení prediktorů a závislé proměnné pro trénink a test dat
X_train, y_train = train[predictors], train['result']
X_test, y_test = test[predictors], test['result']

# standardizace dat - prevence přetrénování
scaler = preprocessing.StandardScaler().fit(X_train)
X_train = scaler.transform(X_train)
X_test = scaler.transform(X_test)
```

B.2 Random forest

V první řadě byly naimportovány potřebné knihovny.

```
# import knihoven
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score, accuracy_score
```

Tvorba modelu s pomocí klasifikátoru *RandomForestClassifier*.

```
# tvorba Random Forest klasifikátoru
rf_classifier = RandomForestClassifier(random_state=42)
```

Dále byl model natrénován a byly vytvořeny predikce na testovacím vzorku dat. Následně byla vyhodnocena přesnost modelu a vygenerována významnost jednotlivých prediktorů.

```
# trénink modelu
rf_classifier.fit(X_train, y_train)

# tvorba predikcí na testovacím vzorku
y_pred = rf_classifier.predict(X_test)

# zhodnocení přesnosti modelu
accuracy = accuracy_score(y_test, y_pred)
print(f"Accuracy: {accuracy:.4f}")

# zobrazení významnosti jednotlivých prediktorů
feature_importance_df = pd.DataFrame({
    'Feature': predictors,
    'Importance': rf_classifier.feature_importances_
})
feature_importance_df = feature_importance_df.sort_values(by='Importance',
ascending=False)
print("\nFeature Importance:")
print(feature_importance_df)

# generace confusion matice
conf_matrix_rf = confusion_matrix(y_test, y_pred)
print("Confusion Matrix for Random Forest:")
print(conf_matrix_rf)
```

B.3 Logistická regrese

V první řadě byly naimportovány potřebné knihovny.

```
# import knihoven
import statsmodels.api as sm
import numpy as np
from sklearn.metrics import confusion_matrix, accuracy_score
```

Přidání konstanty pro stanovení základní pravděpodobnosti výsledku při hodnotách prediktorů nastavených na 0.

```
# přidání konstanty
X_train_sm = sm.add_constant(X_train)
```

Pro vytvoření modelu byla použita třída *Logit*.

```
# nastavení modelu
logit_model = sm.Logit(y_train, X_train_sm)

# trénink modelu
result = logit_model.fit()

# výstup logistické regrese
print(result.summary())
```

```
# výpočet přesnosti modelu
accuracy = accuracy_score(y_test, y_pred)
print(f"Accuracy: {accuracy:.4f}")

# generace confusion matice
conf_matrix = confusion_matrix(y_test, y_pred)
print("Confusion Matrix for Logistic regression:")
print(conf_matrix)
```

Po natrénování modelu bylo potřeba výsledky uložit do nového datasetu *lr_output*, jelikož jsme si model logistické regrese zvolili jako ten, na kterém budeme analyzovat ziskovost sázkových strategií:

```
# přidání sloupců do datasetu test pro analýzu sázkových strategií
test['pl1_prob'] = pred_probs
test['pred_result'] = np.where(test['pl1_prob'] >= 0.5, 1, 0)
test['bookmaker_result'] = np.where(test['pl1_avg_bookmaker_odds'] <
test['pl2_avg_bookmaker_odds'], 1, 0)

# tvorba nového datasetu
lr_output =
test[['id','pl1','pl2','date','pl1_avg_bookmaker_odds','pl2_avg_bookmaker_odds','
result','pl1_prob','pred_result','bookmaker_result']]

# uložení datasetu
lr_output.to_excel('lr_output.xlsx')
```

B.4 Sekvenční neuronová síť

V první řadě byly naimportovány potřebné knihovny.

```
# import knihoven
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense
from sklearn.metrics import confusion_matrix, accuracy_score
```

V další řadě byla definována struktura sekvenční neuronové sítě. V našem případě konkrétně pět skrytých vrstev – u třetí byla zvolena aktivační funkce *RELU*, u zbylých pak *sigmoid*.

```
# definice architektury modelu
model = Sequential([
    Dense(100, activation='sigmoid', input_shape=(X_train_scaled.shape[1],)),
    Dense(50, activation='sigmoid'),
    Dense(25, activation='relu'),
    Dense(10, activation='sigmoid'),
    Dense(1, activation='sigmoid')
])
```

Kompilace modelu byla provedena s optimalizační funkcí *Adam* a ztrátovou funkcí *Binary Cross Entropy*.

```
# kompilace modelu
model.compile(optimizer='adam', loss='binary_crossentropy', metrics=['accuracy'])

# trénink modelu
history = model.fit(X_train_scaled, y_train, epochs=10, batch_size=32,
validation_split=0.2)

# evaluace modelu
y_pred = (model.predict(X_test_scaled) > 0.5).astype(int)
accuracy = accuracy_score(y_test, y_pred)
print(f"Accuracy: {accuracy:.4f}")

# generace confusion matice
conf_matrix_rf = confusion_matrix(y_test, y_pred)
print("Confusion Matrix for Neural Network:")
print(conf_matrix_rf)
```

Příloha C Postup tvorby algoritmů sázkových strategií

Pro tvorbu algoritmů bylo potřeba naimportovat relevantní knihovny a uložit výstup prediktivního modelu do datasetu *df*.

```
import pandas as pd
import numpy as np
df = pd.read_excel("lr_output.xlsx")
```

C.1 Pomocné funkce

V první řadě byla potřeba funkce, která výsledek sázkových strategií zpracuje a předá výstup funkci *display_result*:

```
# Funkce pro zpracování výsledku algoritmů
def process_result(df, func, betting_unit, par2=0, par3=0, save=0):
    # Náhodné zamíchání řádků v datasetu
    shuffled_df = df.sample(frac=1).reset_index(drop=True)

    # Zavolání konkrétní sázkové strategie
    func_output = func(shuffled_df, betting_unit, par2, par3)

    # Tvorba sloupce pro vypočítaný profit každé sázky
    shuffled_df['profit'] = func_output[0]

    # Vytvoření proměnných pro vypsání výsledků
    func_name = func_output[1]
    win_count = func_output[2]
    lose_count = func_output[3]
    win_rate = (win_count / (win_count + lose_count)) * 100
    bet_num = len(df) - np.count_nonzero(shuffled_df['profit'] == 0)
    stakes_sum = bet_num * betting_unit

    # Výpočet celkové sumy profitu
    profit_sum = shuffled_df['profit'].sum()

    # výpočet výnosnosti sázení
    betting_yield = (profit_sum / stakes_sum) * 100

    # pokud je parametr funkce save==1, uložit výsledek do excelu
    if save == 1:
        shuffled_df.to_excel(f"betting_output/{func_name}.xlsx")

    # zavolání funkce display_result pro vypsání výsledku
    display_result(stakes_sum, profit_sum, bet_num, betting_yield, func_name,
win_count, lose_count, win_rate)
```

V případě strategií s negativní progresí byly při každém pokusu řádky v datovém setu náhodně promíchány. Následný kód je pomocná funkce, která zpracuje 100 iterací vybrané sázkové strategie a vypočítá průměrný výsledek použité strategie s negativní progresí. Výsledek je předán funkci *display_result*:

```
def process_martingale_test(df, func, betting_unit, par2=0, par3=0, save=0,
iter_num=100):
    x1, x2, x3, x4, x5, x6, x7 = 0, 0, 0, 0, 0, 0, 0
    for i in range(iter_num):
        # Náhodné zamíchání řádků v datasetu
        shuffled_df = df.sample(frac=1).reset_index(drop=True)

        # Zavolání konkrétní sázkové strategie
        func_output = func(shuffled_df, betting_unit, par2, par3)

        # Tvorba sloupce pro vypočítaný profit každé sázky
        shuffled_df['profit'] = func_output[0]

        # Vytvoření proměnných pro vypsání výsledků
        win_count = func_output[2]
        lose_count = func_output[3]
        win_rate = (win_count / (win_count + lose_count)) * 100
        bet_num = len(df) - np.count_nonzero(shuffled_df['profit'] == 0)
        stakes_sum = bet_num * betting_unit

        # Výpočet celkové sumy profitu
        profit_sum = shuffled_df['profit'].sum()

        # výpočet výnosnosti sázení
        betting_yield = (profit_sum / stakes_sum) * 100

        # Při každé iteraci připočte nové hodnoty do promenných x1-7
        x1 += stakes_sum
        x2 += profit_sum
        x3 += bet_num
        x4 += betting_yield
        x5 += win_count
        x6 += lose_count
        x7 += win_rate

    # Výpočet průměru hodnot vydělením celkového počtu iterací od x1-7
    func_name = func_output[1] + '_(AVG_OF_100_RUNS)'
    avg_stakes_sum = x1 / iter_num
    avg_profit_sum = x2 / iter_num
    avg_bet_num = x3 / iter_num
    avg_betting_yield = x4 / iter_num
    avg_win_count = x5 / iter_num
    avg_lose_count = x6 / iter_num
    avg_win_rate = x7 / iter_num

    # zavolání funkce display_result pro vypsání výsledku
    display_result(avg_stakes_sum, avg_profit_sum, avg_bet_num,
avg_betting_yield, func_name, avg_win_count, avg_lose_count, avg_win_rate)
```

Poslední pomocnou funkcí je funkce, která nám vypíše výstup sázkových algoritmů:

```
# Funkce pro vypsání výsledku strategií
def display_result(stakes_sum, profit_sum, bet_num, betting_yield, func_name,
win_count, lose_count, win_rate):
    print(func_name.upper() + ':')
    print("=========================")
    print("Stakes sum:", round(stakes_sum,1))
    print("Net profit:", round(profit_sum,1))
    print("Yield:     ", round(betting_yield,3), "%")
    print("Total bets:", bet_num)
    print("Bets won:  ", win_count)
    print("Bets lost: ", lose_count)
    print("Win rate:  ", round(win_rate,3), '%')
    print("=========================")
```

C.2 Strategie A a B

Funkce *result_fixed* projde všechny řádky tabulky se zápasy a porovná hodnotu reálného výsledku (0 nebo 1) a predikovaným výsledkem (0 nebo 1), a to buď podle našich predikcí nebo predikcí bookmakerů. V případě shody těchto hodnot se uloží čistý zisk sázky do listu *profit_lst*. V opačném případě se do téhož listu uloží ztráta na sázce. Distinkci použitých hodnot predikcí zvolíme argumentem *model*. Při volání funkce *process_result* jako čtvrtý argument předáváme „bookmaker_result“ pro zvolení predikcí bookmakerů, anebo „pred_result“ pro volbu predikcí modelu logistické regrese.

```
# podle parametru model volba strategie A nebo B
def result_fixed(df, betting_unit, model, par1=0):
    profit_lst = []
    win_count = 0
    lose_count = 0
    func_name = 'lr_result_fixed' if 'pred' in model else 'bm_result_fixed'
    pl_odds = ['pl2_avg_bookmaker_odds', 'pl1_avg_bookmaker_odds']
    for index, row in df.iterrows():
        if row[f'{model}'] != row['result']:
            profit_lst.append(-betting_unit)
            lose_count += 1
        else:
            odds = pl_odds[row['result']]
            profit_lst.append((betting_unit*row[f'{odds}'])-betting_unit)
            win_count += 1
    return [profit_lst, func_name, win_count, lose_count]

# zavolání funkce process_result pro strategii A s inputem 'bookmaker_result'
process_result(df, result_fixed, 500, 'bookmaker_result')

# zavolání funkce process_result pro strategii B s inputem 'pred_result'
process_result(df, result_fixed, 500, 'pred_result')
```

C.3 Strategie C

Funkce *bm_probability_fixed* projde pouze řádky, kde je alespoň jeden z vypsaných kurzu (na hráče 1 nebo hráče 2) menší nebo roven námi stanovenému limitu argumentem *odds_limit*. Vybíráme tak pouze zápasy, kde má jeden z hráčů určitou pravděpodobnost na výhru. Následně se opět porovnává hodnota reálného výsledku zápasu (0 nebo 1) a hodnota predikce bookmakerů. Při výhře sázky se do listu *profit_lst* přičte čistý zisk, naopak při prohře se odečte prohraná suma.

```
def bm_probability_fixed(df, betting_unit, odds_limit, arb=0):
    profit_lst = []
    odds_lst = ['pl2_avg_bookmaker_odds', 'pl1_avg_bookmaker_odds']
    win_count = 0
    lose_count = 0
    for index, row in df.iterrows():
        # Pokud je jeden z vypsaných kurzu menší nebo roven odds_limit
        if row[odds_lst[0]] <= odds_limit or row[odds_lst[1]] <= odds_limit:
            for i in range(2):
                if i == 0:
                    pl = 1
                else:
                    pl = 0
                if row[odds_lst[i]] <= odds_limit:
                    if row['bookmaker_result'] == i and row['result'] == pl:
                        profit_lst.append(-betting_unit)
                        lose_count += 1
                    elif row['bookmaker_result'] == i and row['result'] == i:
                        profit = (betting_unit*row[odds_lst[i]])-betting_unit
                        profit_lst.append(profit)
                        win_count += 1
                    else:
                        profit = (betting_unit*row[odds_lst[i]])-betting_unit
                        profit_lst.append(profit)
                        win_count += 1
                else:
                    continue
        else:
            profit_lst.append(0)
    return [profit_lst, 'bm_probability_fixed', win_count, lose_count]
```

C.4 Strategie D1 a D2

V případě strategie D1 bez stanovení spodní hranice vypsaného kurzu nastavíme parametr *odds_limit* na hodnotu 0. V případě strategie D2 pak na námi zvolenou hodnotu (pro náš příklad konkrétně 1,8). V další řadě ověříme, zda je na řádku naše predikovaná pravděpodobnost výhry u hráče 1 nebo u hráče 2 rovna nebo vyšší námi zvolené procentuální hranici argumentem *prob_limit*. Dále zkontrolujeme, zdali je u hráče, který splňuje předchozí podmínku vypsaný kurz, který je vyšší než kurz definovaný argumentem *odds_limit*. Vybereme tak pouze ty řádky, které tyto podmínky splňují. Následně pak, jako u předchozích strategií, přičítáme výsledky sázek do listu *profit_lst*.

```
def lr_probability_fixed(df, betting_unit, prob_limit, odds_limit):
    profit_lst = []
    odds_lst = ['pl2_avg_bookmaker_odds', 'pl1_avg_bookmaker_odds']
    win_count = 0
    lose_count = 0
    for index, row in df.iterrows():
        # Výběr vítěze
        pl1_bet = row['pl1_prob'] >= 1-prob_limit
        pl2_bet = row['pl1_prob'] <= prob_limit
        pl_bet_lst = [pl2_bet, pl1_bet]
        # Ověření podmínky prob_limit a odds_limit
        if ((pl1_bet and row[odds_lst[1]] > odds_limit)
        or (pl2_bet and row[odds_lst[0]] > odds_limit)):
            for i in range(2):
                if pl_bet_lst[i]:
                    if row['pred_result'] != row['result']:
                        profit_lst.append(-betting_unit)
                        lose_count += 1
                    else:
                        profit = (betting_unit*row[odds_lst[i]])-betting_unit
                        profit_lst.append(profit)
                        win_count += 1
                else:
                    continue
        else:
            profit_lst.append(0)
    return [profit_lst, 'lr_probability_fixed', win_count, lose_count]
```

C.5 Strategie E a F

Vstupní argument *model* slouží pro volbu použitých predikcí – při hodnotě *bookmaker_result* funkce pracuje s predikcemi od bookmakerů; při hodnotě *pred_result* funkce pracuje s námi vygenerovanými predikcemi (viz volání funkce *process_result* níže). Následně pouze aplikujeme systém *Martingale* na každou sázku.

```
# podle parametru model volba strategie E nebo F
def result_martingale(df, betting_unit, model, par1=0):
    profit_lst = []
    original_betting_unit = betting_unit
    win_count = 0
    lose_count = 0
    func_name = ('lr_result_martingale'
                 if 'pred' in model
                 else 'bm_result_martingale')
    pl_odds = ['pl2_avg_bookmaker_odds', 'pl1_avg_bookmaker_odds']
    for index, row in df.iterrows():
        if row[f'{model}'] != row['result']:
            profit_lst.append(-betting_unit)
            betting_unit *= 2
            lose_count += 1
        else:
            odds = pl_odds[row['result']]
            profit_lst.append((betting_unit*row[f'{odds}'])-betting_unit)
            betting_unit = original_betting_unit
            win_count += 1
    return [profit_lst, func_name, win_count, lose_count]

# zavolání funkce process_result pro strategii E s inputem 'bookmaker_result'
process_result(df, result_martingale, 500, 'bookmaker_result')

# zavolání funkce process_result pro strategii F s inputem 'pred_result'
process_result(df, result_martingale, 500, 'pred_result')
```

C.6 Strategie G

Funkce *bm_probability_martingale* vybírá pouze zápasy, kde je absolutní hodnota rozdílu vypsaného kurzu pro hráče 1 a hráče 2 menší než 0,2. Následně pak sázky probíhají stejným stylem jako u ostatních strategií s fixní výší sázek.

```
def bm_probability_martingale(df, betting_unit, arb1=0, arb2=0):
    profit_lst = []
    original_betting_unit = betting_unit
    win_count = 0
    lose_count = 0
    pl_odds = ['pl2_avg_bookmaker_odds', 'pl1_avg_bookmaker_odds']
    for index, row in df.iterrows():
        # Stanovení podmínky vyrovnanosti kurzů (maximální odchylka 0.2)
        if ((row['pl1_avg_bookmaker_odds'] < row['pl2_avg_bookmaker_odds']+0.2)
        and (row['pl1_avg_bookmaker_odds'] > row['pl2_avg_bookmaker_odds']-0.2)):
            if row['bookmaker_result'] != row['result']:
                profit_lst.append(-betting_unit)
                betting_unit *= 2
                lose_count += 1
            else:
                odds = pl_odds[row['result']]
                profit_lst.append((betting_unit*row[f'{odds}'])-betting_unit)
                betting_unit = original_betting_unit
                win_count += 1
        else:
            profit_lst.append(0)
    return [profit_lst, 'bm_probability_martingale', win_count, lose_count]
```

C.7 Strategie H

Podobně jako u strategie G, funkce *lr_probability_martingale* sází pouze na ty zápasy, kde je pravděpodobnost výhry hráče 1 v rozmezí 48–52 %. Následně pak sázky opět probíhají stejným stylem jako u ostatních strategií s fixní výší sázek.

```
def lr_probability_martingale(df, betting_unit, arb1=0, arb2=0):
    profit_lst = []
    original_betting_unit = betting_unit
    win_count = 0
    lose_count = 0
    pl_odds = ['pl2_avg_bookmaker_odds', 'pl1_avg_bookmaker_odds']
    for index, row in df.iterrows():
        # Stanovení vyrovnanosti pravděpodobností (rozmezí 48-52)
        if ((row['pl1_prob'] > 0.48) and (row['pl1_prob'] < 0.52)):
            if row['pred_result'] != row['result']:
                profit_lst.append(-betting_unit)
                betting_unit *= 2
                lose_count += 1
            else:
                odds = pl_odds[row['result']]
                profit_lst.append((betting_unit*row[f'{odds}'])-betting_unit)
                betting_unit = original_betting_unit
                win_count += 1
        else:
            profit_lst.append(0)
    return [profit_lst, 'lr_probability_martingale', win_count, lose_count]
```